MASTER IN COMPUTER SCIENCE

Adaptive antenna arrays in QualNet simulator

Cornet, Sébastien; Dynerowicz, Seweryn

Award date:
2007

Adaptive Antenna Arrays in QualNet Simulator

Sébastien Cornet, Seweryn Dynerowicz

Mémoire présenté en vue de l'obtention du grade de Maître en Informatique

31 Mai 2007

Facultés Universitaires Notre-Dame de la Paix

Abstract

Wireless technology brings many new possibilities to how networks can be used. However many physical phenomenons cause this technology to be less efficient than their wired counterparts. Such problems include Rayleigh Fading and Doppler effect. The problem of frame collision which cause corruption of both frames was solved by the use of a protocol such as *CSMA/CA*. However this protocol works by preventing simultaneous communication on the shared medium, resulting in larger delays, wasted time and bandwidth. The Adaptive Antenna Array is presented as an alternative to the problem of interference, Rayleigh fading and Doppler effect. We present in this thesis the implementation of this antenna model in the QualNet simulator.

Keywords : adaptive antenna array, MANET, AODV, OLSR, rayleigh fading, doppler effect, interference

Les technologies sans-fils amènent de nouvelles possibilités d'utilisation des réseaux. Cependant de nombreux phénomènes physiques entrainent une perte d'efficacité de ces technologies par rapport aux réseaux câblés. Ces problèmes sont par exemple le fading de Rayleigh ou l'effet Doppler. Le problème de collision qui provoque la corruption de trames a été solutionné par des protocoles tels que le *CSMA/CA* pour la norme *802.11*. Cependant ce protocole travaille en empechant des communications simultanées sur le médium partagé ce qui résulte en de plus long délais, du gaspillage de temps et de bande passante. Les antennes adaptatives sont présentés comme une alternative au problème de l'interférence, du fading de Rayleigh et de l'effet Doppler. Nous présentons dans cette thèse une implémentation de ce modèle d'antenne au sein du simulateur QualNet.

Mots-clés : antenne adaptative, MANET, AODV, OLSR, fading de Rayleigh, effet Doppler, interference

Acknowledgements

First of all, i would like to thank Laurent Schumacher (FUNDP) and Yukihiro Kamiya (TUAT) without whom this thesis would not have taken place and for their skills and availability. I deeply thank my parents for their trust and their financial help. I thank my friends for their incentives and all the students from Kamiya Lab for their welcome and assistance during our internship. And last but not least, Seweryn, who greatly helped me for this work.

Sebastien

I would like to thank Kamiya Yukihiro (TUAT) for his presence and guidance during our traineeship and for his outstanding teaching which provided us the required background for the accomplishement of this project. I also deeply thank all the students of Kamiya Lab for their friendly welcome and their help.

In Belgium, I thank Laurent Schumacher (FUNDP) for his invaluable help which allowed us to complete this thesis. I also thank Sebastien, whose inspired ideas were a guiding force for this project.

Je remercie également mes parents et mon frère pour leur soutien et leur aide.

Seweryn

Contents

List of Figures

Chapter 1

Introduction

Nowadays, networking technologies have become part of our everyday life. These technologies have changed the way we work, the way we live and the way we communicate. The emergence of wireless technology enabled to extend the field of application of networking, and it is now possible to use networking in a way that was not possible with wired networks. Ad-hoc networking constitutes one of these new uses. Essentially, an ad-hoc network is an infrastructure-less network, that is, a network which does not contain a single node responsible for all maintenance tasks. All these tasks which are usually handled by a router are taken care of by the peers themselves.

Many research projects have been conducted to find out how each layer of the OSI protocol stack should be adapted. This report focuses on a promising technology located at the physical layer.

The Adaptive Array Antenna is a new type of antenna which is able to receive signals better in environments with much interference. Because of the use of a shared communication medium in wireless standards such as 802.11b, communications can be easily corrupted by interfering transmissions. Several solutions have been proposed such as CSMA/CA. The problem with these traditional solutions is that they try to avoid interference rather than cope with them. In such scenario, there is no possibility for two competing communications to occur at the same time. The AAA tries to solve the problem caused by interference by eliminating or reducing the impact of interference. Consequently, all nodes can communicate with each other without any loss due to interfered signals. This enables to increase the throughput of the network and to suppress the delay in transmissions due to these mechanisms.

Two master thesis have already been realized in this field and proved the effectiveness of AAA. The work presented in this thesis provides a model

which can be used in the QualNet simulator for developping new protocols and testing their effectiveness with AAA. It can also be used to find how each layer of the OSI protocol stack should be designed in order to achieve cross-layer optimization.

The remainder of this report is structured as follows. Chapter 2 covers the theoretical background required to understand how the Adaptive Array Antenna works. The different aspects which influence the environment of the antenna and the optimal weight theory are also presented. Readers familiar with signal processing and AAA theory can safely skip this chapter.
Chapter 3 presents the QualNet tool which is used for the simulations. The simulation engine and the programming interface provided by QualNet are also described.
Chapter 4 focuses on the implementation of Adaptive Array Antenna. We present a brand new antenna model which can be used with any physical layer. We also explain the parameters, simplifications, fading integration and provided statistical results.
Chapter 5 presents the gain and SNR values obtained with the AAA depending on the number of antenna elements and the presence of fading, in order to show that our implementation matches as much as possible the expected behavior of a real Adaptive Array Antenna.
Chapter 6 presents different Java tools developed in order to run batches of simulations, manipulate the results obtained and compute average statistics.
As AAA can change nodes communication behavior, other layers should be tuned to work more efficiently with AAA so, in chapter 7, we present some modifications to PHY and MAC layers which could improve the performance of an Ad-hoc network.
Chapter 8 discusses the results of our static network simulations.
Chapter 9 discusses the results of our mobile network with fading simulations.
Finally, chapter 10 presents our conclusions about this work and AAA in general.

Chapter 2

Theoretical basis

In this chapter, we will present the theoretical basis required to understand how the AAA works. We will go through the representation of signals by using complex numbers. We will then present the AAA mathematical modeling and explain the optimal weight theory. We will mainly focus on the mathematical aspects of the AAA. For a more thorough introduction of signal processing to beginners, the reader may refer to [ ].

2.1 Signal representation

2.1.1 An analogy for signal representation

For the reader to understand the concept of the signal, we will use an analogy. Consider a pond of water in which a stone is thrown. A wave will propagate in a circular way from the point where the stone hit the water. As the wave travels away from the initial point, it will decrease in intensity. A signal emitted by an omnidirectionnal antenna can be assimilated to the wave travelling across the water's surface.

2.1.2 Traditionnal representation

In the information theory, a signal corresponds to a transmission of information over a communication channel. The encoding of the information into the channel is accomplished by modifying the state of that channel. The sequence of states that will be applied to the channel corresponds to the information that is being transmitted. Usually, the encoding of information is accomplished by varying the voltage, current or the electric field strentgh.

Mathematically, we can say that a signal is a function $s(t)$ where t represents the time, and $s(t)$ is the intensity of the disturbance of the medium. Usually, $s(t)$ is defined by a sinusoid curve.

A periodic signal is defined by several parameters :

- Amplitude : the maximum disturbance of the wave during one oscillation.
- Period : the time required for the wave to complete one oscillation.
- Frequency : the number of oscillations during a period of time. Usually measured in Hz (Hertz). A frequency of 1 Hz means that the wave oscillates once every second.
- Wavelength : the distance covered by the wave during one oscillation.
- Phase : at an instant t, the phase is equal to $\phi(t) = (2\pi ft + \theta)$, where f is the frequency and θ is the initial phase shift of the signal.

We can define mathematically a signal wave if we use these parameters in the following way :

$$s(t) = A\cos(2\pi ft + \phi) \tag{2.1}$$

with :

- A : the amplitude of the signal
- f : the frequency [Hz]
- ϕ : the initial phase-shift of the signal [rad]

Equation (2.1) gives us the classical expression of a signal. Note that the cosine function can be easily replaced by a sine function since $cos(x) = sin(\frac{\pi}{2} - x)$

If we consider $A = 1$, $f = 1$ and $\phi = 0$, we obtain the following signal wave :

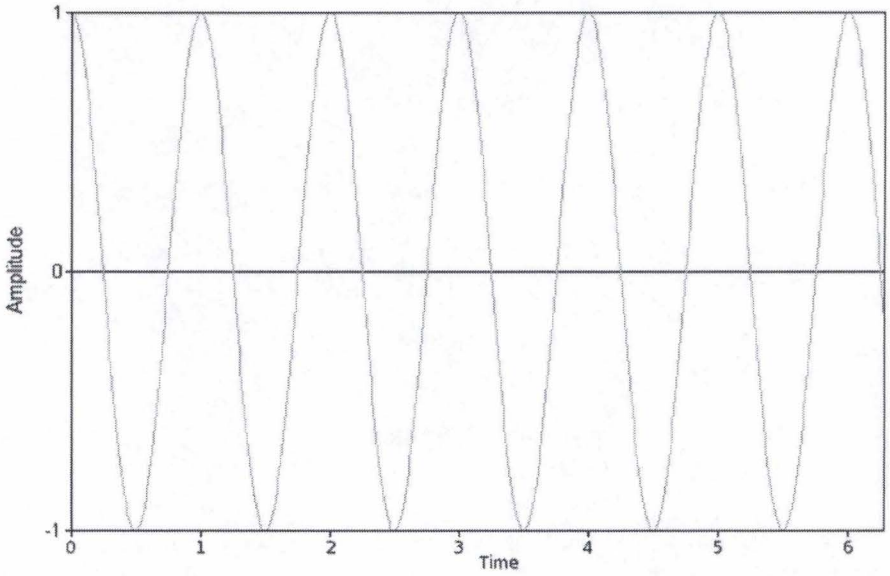

Figure 2.1: Signal wave

which has a period of one second and a frequency of 1 Hz.

The representation given by equation (2.1) contains all the information defining a certain signal. However, because of the use of a *cos* function, this representation is not suited for mathematical manipulations. This is the reason why we try to define another representation which enables to define signals with the same completeness and to express them in a simple way, which is more suited for mathematical manipulations.

2.1.3 Trigonometric representation

It is possible to represent the information contained in (2.1) in vector form.

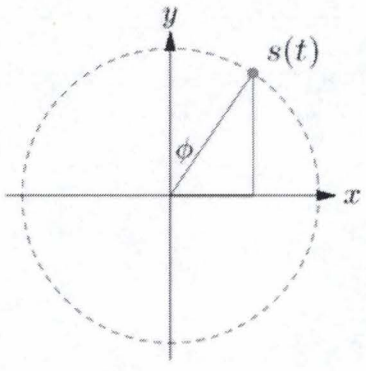

Figure 2.2: Vector representation

In Figure 2.2, the 3 parameters used in equation (2.1) are represented as follows :

- Amplitude : the length of the vector
- Frequency : the number of 2-π phase rotations of the vector during a given period of time. A frequency of 1 Hz means that the vector rotates its phase by 2π once every second.
- Phase : the angle between the vector and the horizontal axis at an instant t

The mapping of the vector on the horizontal and vertical axis is realized by using a cosine and a sine function respectively. We have :

$$x = A\cos\theta \tag{2.2}$$
$$y = A\sin\theta \tag{2.3}$$

We now have a vector representation of a signal. However there is still one step that is required, that is to cast this representation in the complex field.

2.1.4 Analytical signal

The analytical signal is the name used to designate a representation of a signal with the use of complex number. A complex number is a number of the form :

$$z = x + jy \tag{2.4}$$

where x is called the real part and y is called the imaginary part. The number j is an imaginary number for which properties such as $j^2 = -1$ hold.

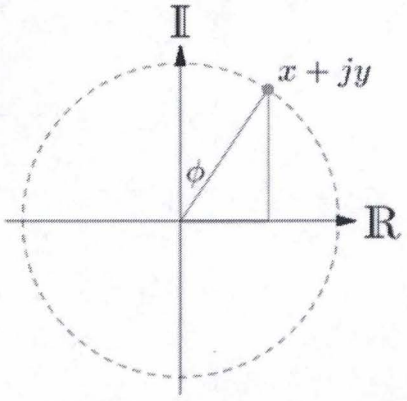

Figure 2.3: Representation of a complex number in the Argand Plane [ ]

If we consider the trigonometric representation presented in the previous section, we can easily show the link with a representation in the complex plane.

By mapping our vector representation described by figure 2.2 to the Argand plane showed in figure 2.3, and considering equations (2.2) and (2.3) we can see that the complex equivalent of the signal will be :

$$z = A\cos\phi + jA\sin\phi \tag{2.5}$$

which we can rewrite as :

$$z = A(\cos\phi + j\sin\phi) \tag{2.6}$$

This is where we will use an important result known as Euler's theorem. This result establishes a link between the trigonometric functions and the complex exponant function :

$$e^{j\phi} = \cos\phi + j\sin\phi \tag{2.7}$$

By using (2.7) in (2.6), we obtain a complex representation of a signal :

$$z = Ae^{j\phi} \tag{2.8}$$

It is interesting to note that we can calculate the amplitude and the power of a signal quite easily :

$$\text{Amplitude} : |z| = \sqrt{x^2 + y^2} = \sqrt{zz^*} \tag{2.9}$$

$$\text{Power} : |z|^2 = x^2 + y^2 = zz^* \tag{2.10}$$

Equation (2.8) represents a signal wave only at one instant. We do not have an equivalence with equation (2.1) which represents the signal according to time. In order to achieve this equivalence, we must complete equation (2.8) by including both frequency and time as follows :

$$z_t = Ae^{j\phi_t} \text{ Where } \phi_t = 2\pi ft + \phi \Rightarrow z_t = Ae^{j(2\pi ft+\phi)} \tag{2.11}$$

2.1.5 Relationship between real signals and analytical signals

We have seen that signals are traditionally represented with equation (2.1) but it is also possible to use the complex representation given by equation (2.8). Because of Euler's theorem, we can write :

$$s(t) = A\cos(2\pi ft + \phi) = Re[Ae^{j(2\pi ft+\phi)}]$$

where $Re[c]$ gives the real part of the complex number c. We can rewrite the rightmost term as :

$$Re[Ae^{j(2\pi ft+\phi)}] = Re[Ae^{j\phi}e^{j2\pi ft}]$$

- $Ae^{j\phi}$ is called the complex envelope
- $e^{j2\pi ft}$ is called the carrier

During this study, we focused our attention on the complex envelope, since it is the part which contains information. The purpose of the carrier part is only to be able to lift up the frequency of the signal in order to achieve better transmission.

The complex representation separates the information from the signal itself, enabling us to reason only on the contents of a signal irrespective of its carrier. More precisely, the phase and the frequency are clearly separated. This aspect will enable us to easily represent the information contained in a certain signal. This topic will be discussed in the next section.

2.1.6 Modulation

If we consider the signal defined so far, we see that what is represented by equations (2.1) and (2.8) is a blank signal. That is, a periodic oscillation of an electric intensity within a communication channel. Such signal is useless if it does not carry any information. The process of encoding information to a signal is called modulation.

There are 3 different modulation types which can be used :

- Frequency Modulation (FM)
- Amplitude Modulation (AM)
- Phase Modulation (PM)

It is important to mention the fact that a modulation transform bit sequences into symbols, while the demodulation transforms symbols into bit sequences.

Frequency Modulation (FM)

As the name implies, the frequency modulation encodes information into a signal by changing its frequency according to the data bit that must be transmitted.
We will use equation (2.1) to draw a signal wave. Consider that, at an instant t, the value of the data bit encoded $d[t]$ is equal to :

- 0 if $f[t] = 1$
- 1 if $f[t] = 4$

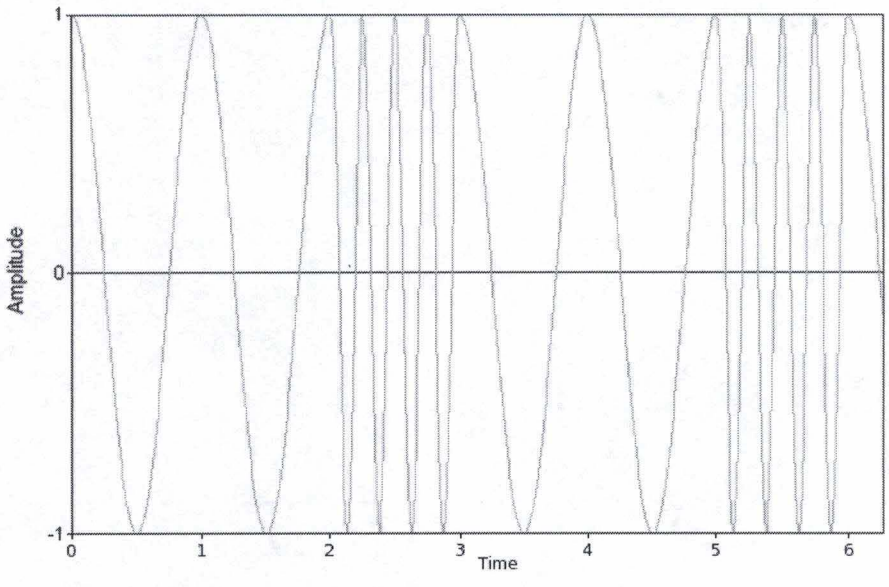

Figure 2.4: Signal representation for the bit sequence '001001' using FM

Amplitude Modulation (AM)

The amplitude modulation encodes information to a signal by modifying its amplitude according to the data bit that must be transmitted.

Consider that, at an instant t, the value of the data bit encoded $d[t]$ is equal to :

- 0 if $A[t] = 1$
- 1 if $A[t] = 0.5$

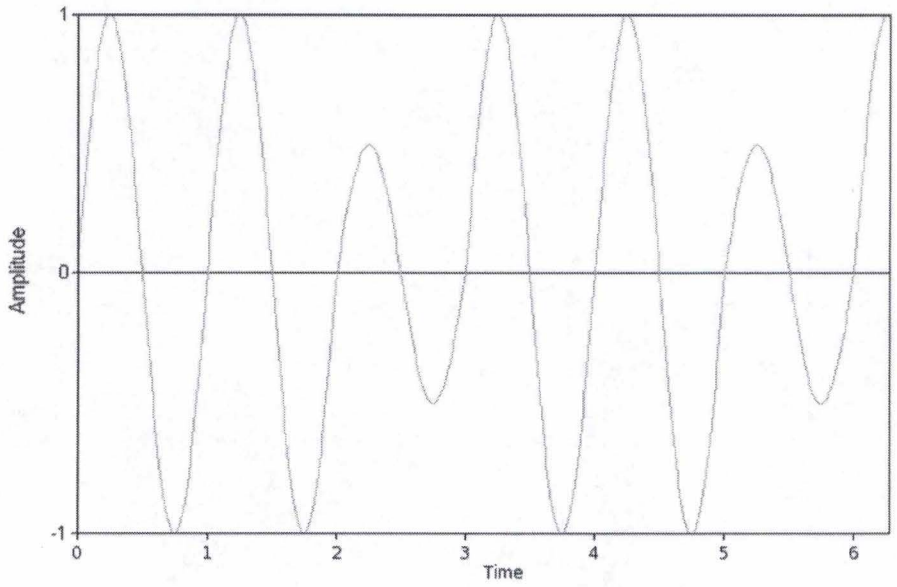

Figure 2.5: Signal representation for the bit sequence '001001' using AM

It is interesting to note that AM signals are more sensitive to ambiant noise than FM signals. This is because the ambiant noise can easily lower the amplitude. If we reduce the amplitude of the signals represented on figure (2.4) and (2.5) by 0.5, we see that the the FM signal will be still be interpreted correctly, because the frequencies will not be changed, whereas the AM signal will be interpreted incorrectly, because the amplitudes of 1 will be reduced to 0.5, and the amplitudes of 0.5 will be nullified, leading to a misinterpretation of the original signal as a sequence of '1'.

Phase Modulation (PM)

The phase modulation is the most important and basic modulation in digital communications. It encodes information into a signal by shifting the phase of a signal according to the data bit that must be transmitted.

Consider that, at an instant t, the value of the data bit encoded $d[t]$ is equal to :

- 0 if $\phi[t] = 0$
- 1 if $\phi[t] = \pi$

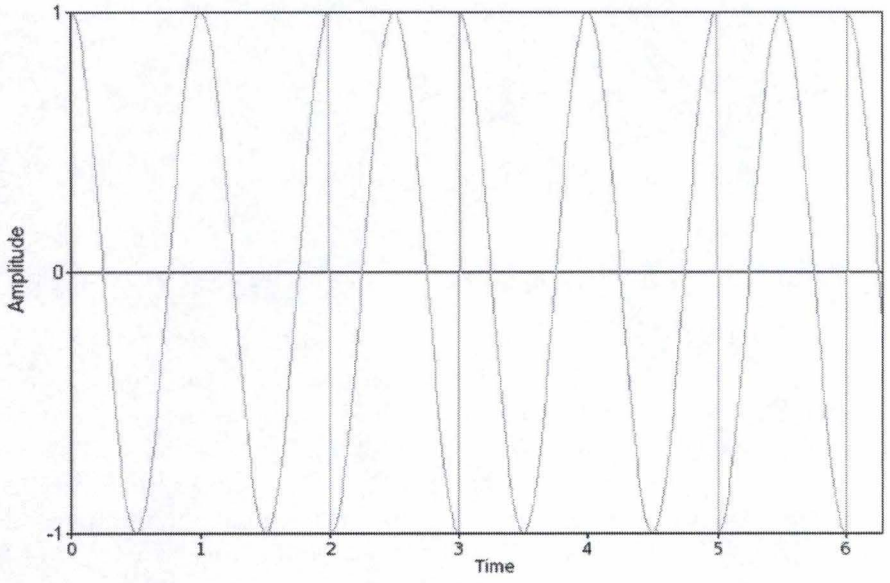

Figure 2.6: Signal representation for the bit sequence '001001' using PM

In the case of digital communication systems, the modulation process is called shift keying. The basic phase shift keyings are Binary PSK (1 bit per symbol), Quadrature PSK (2 bits per symbol) and 8-PSK (3 bits per symbol).

In order to be able to simulate transmission of signals, we need to be able to represent the shift keying in the complex field. Since the phase is represented by the angle of the vector, we can represent the symbols of a PSK by placing them uniformely on a circle centered at (0,0). Each symbol is placed with a $\frac{\pi}{N}$ angle difference, where N is the number of symbols in the modulation. We obtain patterns which are called constellation diagrams.

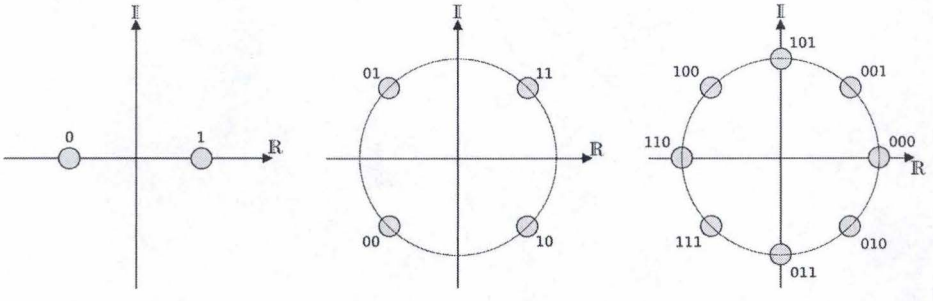

Figure 2.7: Basic PSK constellation diagrams with a Gray coding [ ]

Usually, PSK modulations are used to encode at most 3 bits of data per symbol. This is because the higher the number of bits encoded into a symbol, the more the shift keying is sensitive to noise because a high number of symbols require a small angle between each of them. This is the reason why Quadrature Amplitude Modulation (QAM) type modulation are used to encode more than 3 bits per symbol. In such modulation, both phase and amplitude are modified to encode information. In the case of a 16-QAM (16 symbols containing 4 bits), the symbols are placed on 3 concentric circles, with the outmost one having a radius, and therefore an amplitude of 1. The two inner circles have amplitudes which allow a uniform placement of the symbols.

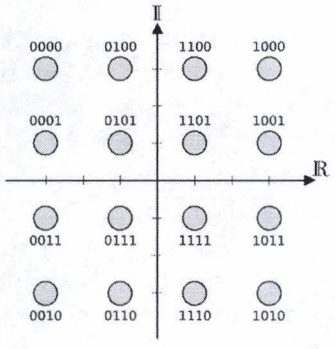

Figure 2.8: Rectangular 16-QAM constellation diagram with a Gray coding [ ]

2.1.7 Gray coding

An interesting aspect of the constellation diagrams is that they follow a Gray coding. It means that any two adjacent symbols only differ by one bit. This property garantees that, for a constant interference and a constant noise, whenever a symbol is disturbed, the number of bits that are misinterpreted will always be same whatever symbol was disturbed. Typically, the BER for a transmission with a specific modulation will be minimum if a Gray coding is used. Figure (2.9) shows the BER level for 5 possible symbol placement for a QPSK modulation. We only represented 1 Gray-coded QPSK modulation, since all Gray-coded modulations yield similar results. The same goes for the non Gray-coded modulations.

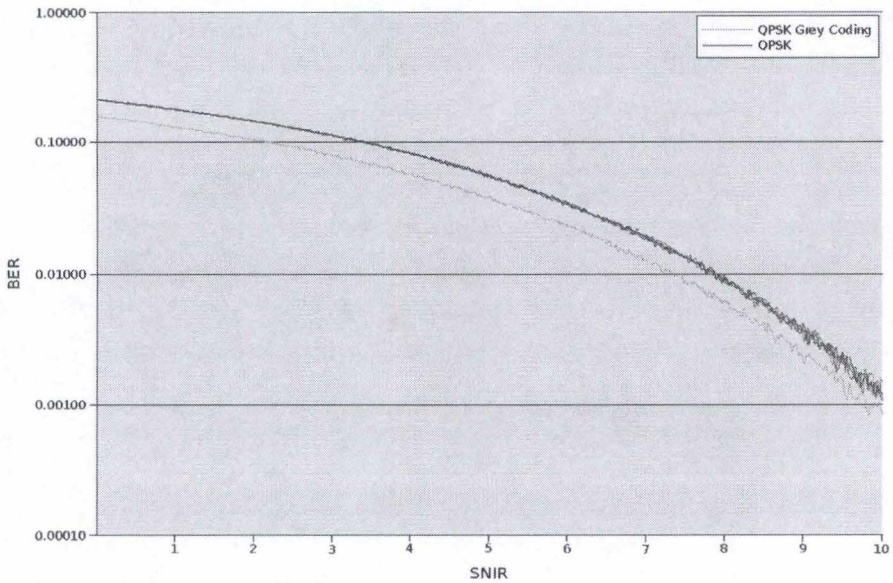

Figure 2.9: Comparison of BER level for 4 QPSK symbol placement. The lowest curve represent the BER level for a QPSK modulation with a Gray coding. The other curves represent the other non-Gray coded modulations.

2.2 Mathematical formulation of AAA principle

In order to understand how the AAA works, we need to define several components. First, we will express the input of an antenna. This input represent what arrives at the antenna elements. We will then formulate the output of the antenna based on the input. We will then show how an important index, the Signal-to-Noise and Interference Ratio (SNIR) can be calculated from the output.

The main idea behind an AAA is to combine several omnidirectionnal antennas. The output of the antenna will be a weighted sum of the outputs of the different elements. By changing the weights, it is possible to increase or reduce the antenna gain in a certain direction. The gain is a measure expressed in dB, which represents the ability of the antenna to amplify or attenuate the signals coming from a certain direction. Therefore, the modification of the weights will make the antenna focus its attention in a specific direction while ignoring or reducing signals coming from another direction. This is the reason why the AAA can be described as a spatial filter. The placing of the elements can be done in many ways. In our implementation, we only used linear and circular placement of the elements. This can however be easily extended to any shape.

2.2.1 Plane wave

The assumption of plane wave is an important concept which is used in the simulation. It allows us to simplify the calculations, especially the estimation of the steering vector which will be explained later.

Consider a receiver Rx and a transmitter Tx. Since electro-magnetic waves propagate spherically, if Rx and Tx are close to each other, the Direction of Arrival (DoA) of the signal will be different for each element of the receive antenna. The idea behind the plane wave assumption is to consider that Tx is far enough from Rx, so that the DoA of a signal is the same for all elements.

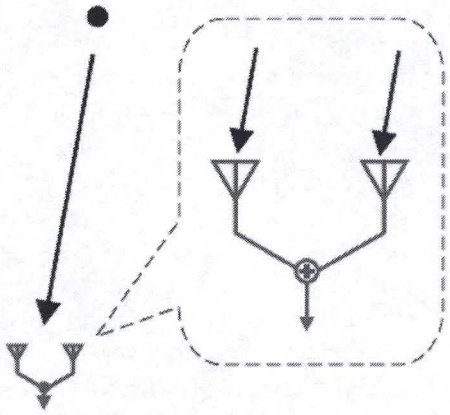

Figure 2.10: Plane wave assumption

2.2.2 Steering vector

Since we have several elements which are separated by a certain distance, the phase of an incoming signal will be different at each of these elements. As we have mentionned before, we consider the plane wave assumption. We will now show how it is possible to compute the phase-shifts according to a certain DoA.

Uniform Linear Array

In the case of a Uniform Linear Array (ULA), the elements are placed on a line and are separated by a distance d.

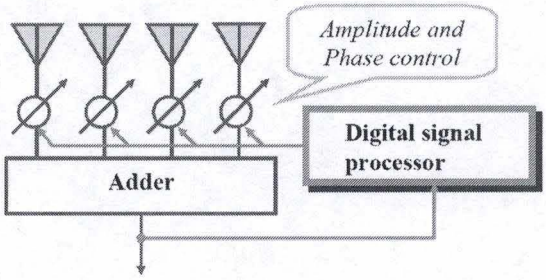

Figure 2.11: Linear element antenna

A signal is arriving from a direction θ. We consider the leftmost element as the reference. This means the phase is considered to be equal to 0 at that

element. The following figure shows how we can deduce the phase-shifts according to a certain DoA.

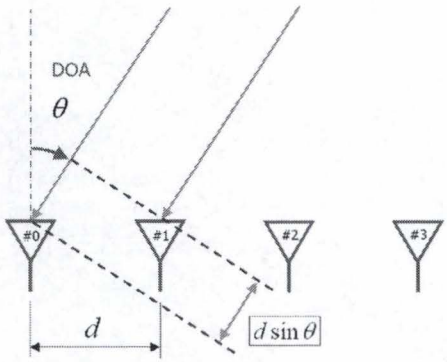

Figure 2.12: Expression of the distance

From Figure 2.12, we can see that we must evaluate the phase-shift caused by the additionnal propagation of the wave by a distance $d\sin\theta$. We now need to express the phase-shift according to a travelled distance x.

$$\phi_{add}(x) = \frac{2\pi x}{\lambda}$$

If we use this definition of the distance, we obtain the expression of phase-shift between two elements :

$$\frac{2\pi}{\lambda} d\sin\theta$$

Since we focus on the phase-shift between element 0 and element 1, we obtain :

$$-\frac{2\pi}{\lambda} d\sin\theta$$

If we want to generalize this formula to calculate the phase-shifts of an N-element antenna, we might write :

$$\forall n, 0 < n \leq N-1 : \phi_n(\theta) = -\frac{2\pi}{\lambda} nd\sin\theta$$

Since we are using complex numbers for the simulation, we need to rewrite the steering vector by using complex envelopes.

$$\mathbf{a}(\theta) = \begin{pmatrix} 1 \\ e^{-j\frac{2\pi d}{\lambda}\sin\theta} \\ \vdots \\ e^{-j(N-1)\frac{2\pi d}{\lambda}\sin\theta} \end{pmatrix}$$

where d is the distance between the elements, λ is the wavelength and N is the number of elements. In this representation, the first line represents the phase-shift at the first element. It is equal to 1 since the first element is considered to be a reference. Each successive line can be obtained by multiplying the previous line by

$$e^{-j\frac{2\pi d}{\lambda}\sin\theta} \tag{2.12}$$

Uniform Circular Array

We have seen how to represent the phase shifts for a ULA. However, in an ad-hoc network, a user does not know *a priori* where the node he or she wants to communicate with is located. Linear antennas suffer from the symmetry of their pattern as two opposite DoA (giving the same angle because of the symmetry) will have the same steering vector, so it can confuse the linear antenna.

When modelling a circular antenna, we consider that all elements are equally distant on a circle of radius r. If we consider the center of that circle as the reference, we can easily express the phase shift at each element.

In this case, the distance can be written as :

$$-\frac{2\pi}{\lambda}r$$

We can then formulate the steering vector itself :

$$\mathbf{a}(\theta) = \begin{pmatrix} e^{-j\frac{2\pi}{\lambda}r\cos\theta} \\ \vdots \\ e^{-j\frac{2\pi}{\lambda}r\cos(\theta-\frac{2\pi(N-1)}{N})} \end{pmatrix}$$

2.2.3 Input of the antenna

We can now define the input of the antenna at an instant t by :

$$\mathbf{x}(t) = s(t)\mathbf{a}(\theta) + \mathbf{n}(t) \tag{2.13}$$

in which s is a vector containing complex numbers which represents the transmitted signal in the form of a sequence of complex envelopes. By multiplying it by the steering vector, $\mathbf{a}(\theta)$, we modify the phase according to the element which receives the signal. $\mathbf{n}$ is the noise vector. In our study, Additive White Gaussian Noise is used since it is the traditionnal way to simulate the ambiant noise. $\mathbf{x}$ corresponds to the input vector of the antenna (i.e. what each element receives), it is a N×L matrix, where N is the number of elements, and L is the number of symbols in the signal.

2.2.4 Output of the antenna

Now that we have defined the input of the antenna at an instant t, we can define the output of the antenna at the same instant :

$$y(t) = \mathbf{w}^H \mathbf{x}(t) \tag{2.14}$$

where $\mathbf{x}$ is the input of the antenna and $\mathbf{w}^H$ is the Hermitian of the weight vector. This equation simply means that the output of the antenna is the sum of the inputs, weighted by the coefficients of $\mathbf{w}^H$. Since $\mathbf{x}$ is a N×L matrix and $\mathbf{w}^H$ is a 1×N vector, $\mathbf{y}$ will be a 1×L vector which corresponds to the symbols the AAA claims to have received from the channel.

2.3 Theory of optimal weights

We have defined so far what the AAA receives and what it outputs. However, the output depends on the weight vector that is used. The problem we need to solve can be stated as follow : how do we compute the weights that will be able to increase the power of a signal coming from a specific direction, which is the desired signal, while decreasing the power of signal(s) coming from other directions, which are interfering and are therefore undesired signals. If we take a good look at equation (2.13), we see that we can use the steering vector obtained from the DoA of a desired signal to align the phases, in other words to cancel the phase differences, of a desired signal. If the phases of the different signals are perfectly aligned, the antenna will output a powered up signal.

Mathematically, the output of the signal is :

$$y(t) = \mathbf{w}^H \mathbf{x}(t)$$

which can be rewritten as :

$$y(t) = \mathbf{w}^H [\mathbf{a}(\theta)\mathbf{s}(t) + \mathbf{n}(t)]$$

If we consider that $\mathbf{w} = \mathbf{a}(\theta)$, we obtain

$$\begin{aligned} y(t) &= \mathbf{a}^H(\theta)[\mathbf{a}(\theta)\mathbf{s}(t) + \mathbf{n}(t)] \\ &= \mathbf{s}(t)\mathbf{a}^H(\theta)\mathbf{a}(\theta) + \mathbf{a}^H(\theta)\mathbf{n}(t) \end{aligned}$$

In the last equation, we will focus on the product $\mathbf{a}^H(\theta)\mathbf{a}(\theta)$. For an antenna with N elements, this part can be rewritten as :

$$\begin{aligned} \mathbf{a}^H(\theta)\mathbf{a}(\theta) &= \sum_{n=0}^{N-1} e^{-j\theta_n} e^{j\theta_n} \\ &= \sum_{n=0}^{N-1} 1 \\ &= N \end{aligned}$$

Therefore, the output of the antenna will be

$$y(t) = N * s(t) + \mathbf{a}^H(\theta)\mathbf{n}(t)$$

In which the power of the signal has been multiplied by N

In the case of an interfering signal, we will try to accomplish the opposite, that is, to decrease the power of the signal. Since two out-of-phase signals cancel each other, the easiest way to decrease the power is to shift the phases even more so that they cancel each other. If we can find a certain weight vector $\mathbf{w}^H$ such that

$$\mathbf{w}^H\mathbf{a}(\theta) = 0$$

the output of the antenna will be canceled.

2.3.1 Wiener solution

We will first refine the definition of the input of the antenna, by including both desired and interfering signals :

$$\mathbf{x}(t) = s_D(t)\mathbf{a}(\theta_D) + \sum_{i \in I} s_i(t)\mathbf{a}(\theta_i) + \mathbf{n}(t) \tag{2.15}$$

where $\mathbf{s}_D$ is the desired signal which is arriving from a direction θ_D. The set I contains all the signals that interfer with $\mathbf{s}_D$ along with their DoA.

Let $\mathbf{X}$ be the input matrix of the antenna. The equation we must solve is :

$$\mathbf{X}\mathbf{w}^* = \mathbf{r}$$

For this equation, the contents of $\mathbf{X}$ can be calculated and $\mathbf{r}$ is known. We can multiply both sides by $\mathbf{X}^H$:

$$\mathbf{X}^H\mathbf{X}\mathbf{w}^* = \mathbf{X}^H\mathbf{r} \tag{2.16}$$

The so-called *correlation matrix*

$$\mathbf{R}_{xx} = \mathbf{X}^H\mathbf{X}$$

is an important one in digital signal processing, since it defines the correlation between all the elements of the array, for each symbol that is received.

We can rewrite equation (2.16)

$$\mathbf{R}_{xx}\mathbf{w}^* = \mathbf{X}^H\mathbf{r}$$

which we can transform into :

$$\mathbf{w}^* = \mathbf{R}_{xx}^{-1}\mathbf{X}^H\mathbf{r}$$

which gives us Wiener solution :

$$\mathbf{w}^H = (\mathbf{R}_{xx}^{-1}\mathbf{X}^H\mathbf{r})^T \tag{2.17}$$

It is important to note that if we use the weight vector defined by equation (2.17), we will obtain a pattern with a correct form, but which is smaller than the optimal pattern. Therefore, we must normalize $\mathbf{w}$ to obtain the optimal pattern. To normalize the weight vector, we compute its norm and then divide the vector by this scalar value.

$$|\mathbf{w}| = \sqrt{\sum_i^N w_i\bar{w}_i} \tag{2.18}$$

$$\mathbf{w} = \frac{\mathbf{w}}{|\mathbf{w}|} \tag{2.19}$$

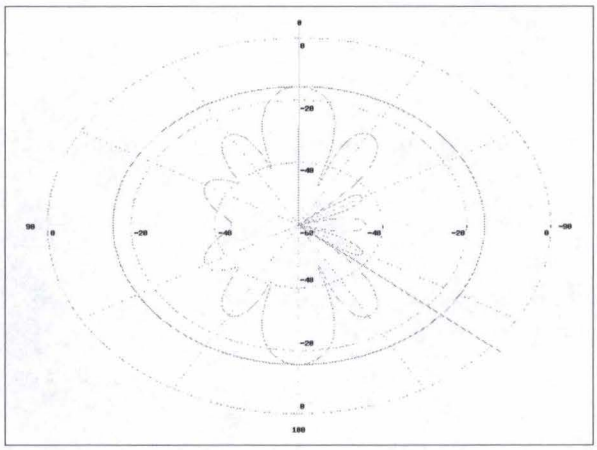

Figure 2.13: Pattern without normalizing the weight vector

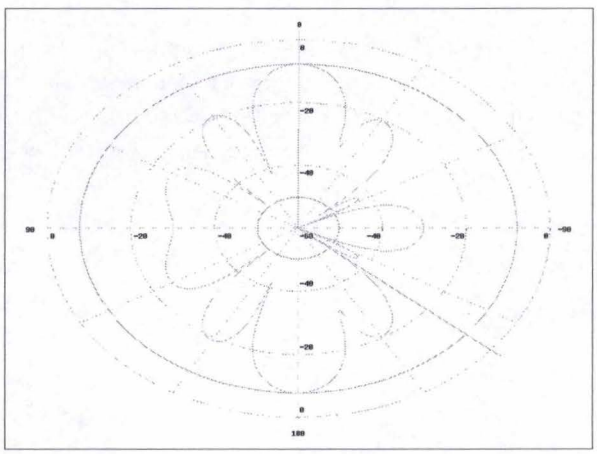

Figure 2.14: Pattern with a normalized weight vector

Figures 2.13 and 2.14 show the patterns obtained by Wiener solution for a desired signal with a DoA of 0 degrees (represented by a blue line) and one interfering signal with a DoA of -130 degrees (represented by a red line). The antenna is a linear array with 6 elements, a BPSK modulation is used and the preamble is 256 bits long. The upper pattern is obtained without normalizing the weight vector $\mathbf{w}$. The lower pattern is obtained by normalizing the weight vector $\mathbf{w}$. The blue and red circles indicate the level

of the gain for the desired and interfering signals respectively.

2.3.2 Minimum Mean Square Error

We have expressed simply in the Section 2.3 how we can increase or decrease the power of a signal. However we showed the two distinct cases separately. This is unfortunately not a useful solution since, in practice :

- Interfering signals will arrive during the reception of a desired signal
- DoA of a signal is unknown

The problem of the DoA can be solved by the use of algebraic methods such as eigenvectors. It will not be discussed in this report. Additionnal information can be found in [ ].

Since we have to deal with a desired signal and one or several interfering signals at the same time, we will use the MMSE theory so that the antenna will increase the power of the desired signal at a maximum, while keeping the power of interfering signals as low as possible.

MMSE is a mean to solve traditionnal optimization problems. Given an error indicator

$$e(t) = r(t) - y(t)$$

where $r(t)$ is the reference signal, that is the signal which should be received, and $y(t)$ is the output of the antenna. The idea of MMSE is to compute w^H in order to minimize $e(t)$. This is a point where we come across one of the problems that must be solved in order to be able to use an AAA in practice : the preamble distribution. In order for an antenna to receive correctly the desired signal, it must know at least its beginning. The idea is to distribute preambles among the nodes, such that whenever a node A wants to communicate with a node B, A will insert the correct preamble in front of the frame it wants to transmit, so that B will be able to focus on the particuliar signal coming from A. The problem of preamble distribution will not be covered in this report.

2.3.3 Least Mean Square algorithm

The LMS algorithm is a simple solution to solve the optimal weight problem (see Section 2.3.2). It performs an iterative computation of the weight vector, by using an approach similar to the steepest descent method. In this section, we simply describe the algorithm. A more thorough derivation can be found in [ ]

Let us suppose we have a reference signal r, an antenna input $\mathbf{x}$. The LMS algorithm will proceed as follows :

1. Set all the elements of the weight vector to 0

2. Set $m = 1$

3. Calculate the output of the antenna $y(m) = \mathbf{w}(m)^H \mathbf{x}(m)$

4. Calculate the error between the output and the desired signal : $e(m) = r(m) - y(m)$

5. Update the weight vector : $\mathbf{w}(m+1) = \mathbf{w}(m) + \mu \mathbf{x}(m) e^*(m)$

6. Increase m by one and go back to step 3

The parameter μ is called the step size. It will determine how fast the LMS will converge to the correct value. The higher μ is, the faster the convergence but the solution will evolve in an unstable way. It is important to note that $0 < \mu \leq 1$

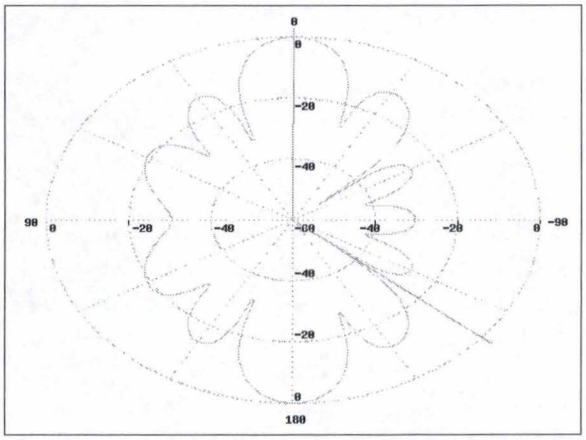

Figure 2.15: Pattern obtained by the LMS algorithm. A desired signal is arriving from 0 degree and an interfering signal is arriving from -130 degrees.The antenna is a linear array with 6 elements, a BPSK modulation is used and the preamble is 256 bits long. The step size is 0.1.

2.4 Multipath fading

So far, we have only considered a simplified environnement. The signals arrived directly at the receiver, and were only modified according to the position of the elements of the antenna. There are several additionnal modifications that are applied by the environnement to a travelling wave. The first one is called fading.

In classical environnement, a travelling wave can be subject to many phenomenons such as reflection, scattering of diffraction. The result of these phenomenons is that several copies (rays) of an identical original signal arrive at the receiver. Since each of these rays followed a different path, their phases, amplitudes and DoAs will be different, hence the name *multipath fading.*

The fading used in our simulation is Rayleigh fading, which is a specific case of Ricean fading. The latter consider there is one direct path (LOS or Line-Of-Sight) between the communicating terminals, while the former models typical environment in mobile wireless communication systems where there is no Line-Of-Sight between the communicating terminals.

If we consider a transmitter who wants to send a specific signal $\mathbf{s}$ over a Rayleigh fading channel. The input of the antenna at the receiver can be written as :

$$\mathbf{x}(t) = \sum_{m=0}^{\mathbf{M}-1} \mathbf{a}(\theta_m)\mathbf{s}(t)e^{j\varphi_m} + \mathbf{n}(t) \tag{2.20}$$

where, $\mathbf{M}$ is the number of rays arriving at the receiver. The Rayleigh phase-shift, φ_m, is chosen randomly between 0 and 2π. Since the different versions of the signal have travelled along different paths, the DoA of a ray (θ_m) can be randomly chosed between 0 and 2π. A more thorough explanation on Rayleigh distribution and the reason why φ_m follow a uniform distribution can be found in [ ].

2.5 Doppler effect

Another aspect which has not been considered so far is the mobility. In the real world, users tend to move in their environnement. During a reception, if the transmitter is moving, the signal received will be changed according to the movement. This effect is similar to the one we witness when an ambulance passes near us. The siren will start with a higher pitch than its pitch when it is stationnary. The pitch will increase as the ambulance approaches and will then gradually decrease as the ambulance moves away. This is simply explained by the fact that as the ambulance approaches, the frequency of the signal is increased since the source is approaching towards us. It is known that high-frequency sounds have a higher pitch. As the

ambulance moves away, the frequency of the signal decreases, causing a lower pitch. It is also possible to express the Doppler effect as a phase-shift.

If we consider a terminal, which is moving at velocity v, towards a direction ψ. The phase-shift resulting on a signal with a DoA ϕ_m can be written as :

$$\frac{2\pi}{\lambda} vt \cos(\psi - \phi_m) \tag{2.21}$$

In order to combine Rayleigh fading and Doppler effect within a single formula, we can simply multiply the signal by the sum of the Rayleigh fading and Doppler effect phase shifts. The input of the antenna will be :

$$\mathbf{x}(t) = \sum_{m=0}^{M-1} \mathbf{a}(\theta_m)\mathbf{s}(t)e^{j\xi_m(t)} + \mathbf{n}(t) \tag{2.22}$$

$$\xi_m(t) = \frac{2\pi}{\lambda} vt \cos(\psi - \phi_m) + \varphi_m \tag{2.23}$$

In equation (2.23), φ_m is the phase-shift caused by rayleigh fading.

2.6 SNIR calculations

The Signal-to-Noise and Interference Ratio, or SNIR, is a very important indicator in wireless communications. It is calculated with :

$$SNIR = \frac{P_s}{P_i + P_n} \tag{2.24}$$

where P_s is the power of the signal, P_i is the power of the interference(s) and P_n is the power of the noise, for which the values is given by

$$P_s = E\left[|s(t)|^2\right] \left|\mathbf{w}^H \mathbf{a}(\theta_s)\right|^2 \tag{2.25}$$

$$P_i = \sum_{i \in I} E\left[|s_i(t)|^2\right] \left|\mathbf{w}^H \mathbf{a}(\theta_i)\right|^2 \tag{2.26}$$

$$P_n = E\left[|n|^2\right] \mathbf{w}^H \mathbf{w} \tag{2.27}$$

where E[.] is the ensemble average.

Unfortunately, this formula is not suitable under Rayleigh fading environment. Figure (2.16) shows the pattern obtained from Wiener solution under Rayleigh Fading. As we can see, the antenna does not orientate its attention towards a specific direction. This is because, under Rayleigh fading, several copies of the desired signal can arrive from several directions, confusing the antenna. The formula for the SNIR described above, calculate the power of the desired signal and the interference according to the gain for their DoA. Because of the shape of the pattern, the value P_d will be close to the value P_i, resulting in a SNIR close to 1.

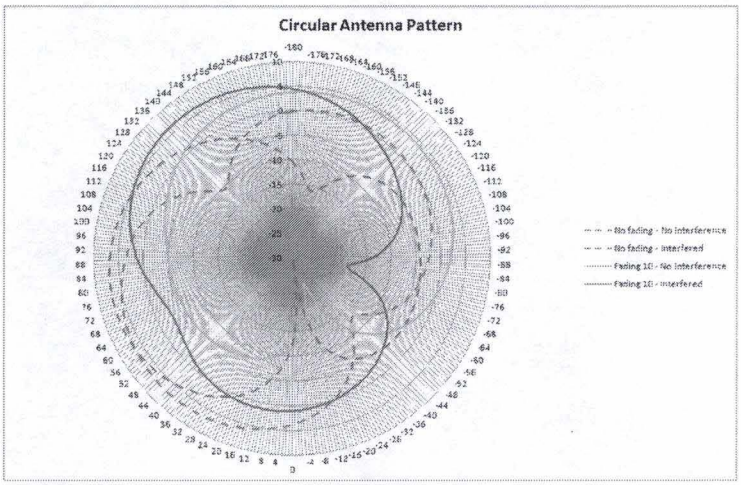

Figure 2.16: Antenna Pattern for a desired signal DoA of 56 degrees and an interfering signal DoA of -6 degrees under Rayleigh fading

We need to use a formula that does not simply use the Gain for the DoA, but the information contained within the signals :

$$SNIR = \frac{|\rho|^2}{1 - |\rho|^2} \tag{2.28}$$

$$\rho = \frac{E\left[\mathbf{x}(t)r^*(t)\right]}{\sqrt{E\left[|\mathbf{x}(t)|^2\right] E\left[|r(t)|^2\right]}} \tag{2.29}$$

where $\mathbf{x}(t)$ are the outputs of the elements and $r(t)$ is the reference signal.

2.7 Gain computation

As we will expose in the following chapter, QualNet requires the value of the gain for each received signal. This gain should therefore be derived. It represents the ability to amplify a desired signal and/or attenuate an interfering one. Defining

$$P(y) = \sum^{Symbols} \sum^{N} |y|^2$$

as the power of received signal, the gain for desired signal with AAA

using the computed weights $\mathbf{w}$ writes

$$Gain_s = \frac{P(s(t)\mathbf{w}^H\mathbf{a}(\theta_s))}{\frac{P(s(t)\mathbf{a}(\theta_s))}{N}} \geq 0 \tag{2.30}$$

whereas the attenuation of the interference, actually a negative gain is given by

$$Gain_i = \frac{P(s_i(t)\mathbf{w}^H\mathbf{a}(\theta_i))}{\frac{P(s_i(t)\mathbf{a}(\theta_i))}{N}} \geq 0$$

This formula computes the antenna gain by dividing the power of output signal (after filtering) by the power of input signal. See Chapter 5 for plots of AAA gain.

Chapter 3

QualNet

This chapter presents the QualNet tool which was used for the simulations. The main purpose is to explain to any new user of QualNet how the simulator works, how to add new types of *layer* and how to run simulations. The whole setup process is clearly described in the Installation Guide provided with QualNet and will not be explained here.

3.1 Introduction to QualNet

The QualNet simulator, is a commercial network simulator suite, developed by Scalable Network Technologies[1]. It was formely based on the GloMoSim simulator[2] (itself using the Parsec framework[3]), written in C/C++. The latest version 4.0 fully supports parallel execution of simulation with multi-core processors which is not the case with the version 3.9.5 that we used for this work. The simulator is discrete-event driven.

3.2 Structure of QualNet

The QualNet simulator is composed of 3 parts : a license server, a simulation engine and a Graphical User Interface (GUI).

Whenever a simulation or the GUI is executed, the instance of the simulation engine or the GUI will attempt to contact the license server in order to validate the action the client wants to perform. If the license is not valid for that action (for example running 3 or more simulations at the same time) or if the licence server cannot be contacted, the instance will stop. Therefore, it is important for the server to be reachable from the client through the network.

[1] http://www.scalable-networks.com/
[2] http://pcl.cs.ucla.edu/projects/glomosim/
[3] http://pcl.cs.ucla.edu/projects/parsec/

3.3 Simulation engine

In order to run a simulation, it is necessary to write a configuration file for the experiment. The simulation engine will read such file to setup the experiment's environment. A typical configuration file contains : the size of the environment, the number of nodes and their positions in the environment, the definition of the OSI Layer Model used by the nodes (either generic for all nodes or specific for each node) and all parameters required by each layer of the OSI Model.

The simulation engine will perform an initialization of the environment by using the definitions read in the configuration file. Afterwards, it will start the simulation itself by starting the internal clock. This simulation internal clock is used to sort, schedule and process events that happen in the simulation. For example, when a wireless node emits a signal, the other nodes will receive the signal a little bit later, after the propagation delay, that is to say the time required for the wave to travel through the medium to the receiver. The signal will finally arrive at the exact time depending on the distance between emitter and receiver.

As shown in figure 3.1, the OSI protocol stack is respected in QualNet simulator, they add an *antenna model* below the *PHY layer* to represent the different types of antenna (steerable, patterned, onmidirectional) which can be used with the simulator, we will explain this antenna layer in the Chapter 4 with our implementation of the *AAA*.

The protocol layers behave as specified by the OSI model. For example, during an FTP connection between wireless nodes, the client node's FTP layer will generate a message representing the packet that an FTP client sends to the FTP server. Afterwards it will transmit it to the *Transport* Layer, which will in turn transmit it to the *Network* Layer and so on. The *Physical* Layer will release the signal to the *Communication* Channel (called *Propagation* layer in QualNet) which will pass it to the *Physical* Layer of the different nodes which should receive the packet (which are in emitter's range). The *Physical* Layer will then handle the received signal as a packet and will forward it to the upper layer and so on up to the *Application* layer (FTP protocol in this case).

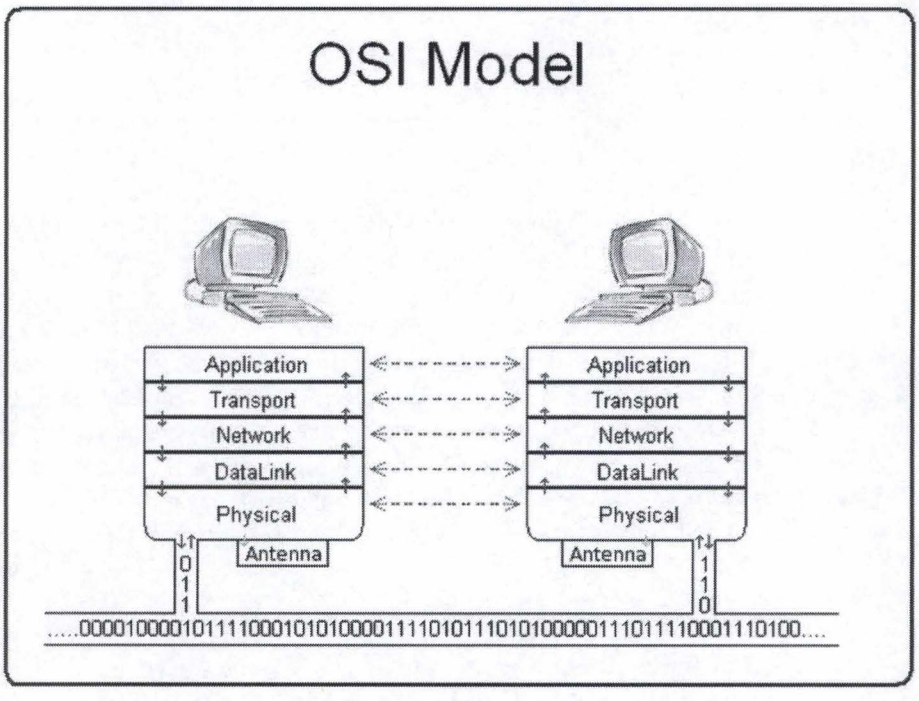

Figure 3.1: Network Stack in QualNet

3.4 Graphical User Interface

The GUI is intended to simplify the definition of the configuration files as well as the interpretation of the results.

3.4.1 Scenario designer

Writing a complete configuration file can be very tedious, especially when one wants to simulate the behavior of a network containing several nodes, each of them having a specific OSI configuration. The scenario designer is a useful tool to create a new simulation experiment by specifying many parameters: position of the nodes (manual, randomly choosen), protocol stack (802.11, 802.16, CSMA/CA, ALOHA, ETHERNET, IPv4, IPv6, RIP, AODV,...) and applications that will communicate with each other (FTP, traffic-gen, VoIP, etc...).

3.4.2 Animator

The Animator is used to run simulations. It provides a graphical view of the environment, the positions of the nodes and different aspects of a simulation like the transmission of broadcast signals, unicast signals and the movement of the nodes. However, it should not be used for large scale simulations since the view is not clear when there are too many nodes. Also the computation speed drops significantly when the GUI is used. Therefore, the Animator should only be used for small simulations or for demonstration purposes.

3.4.3 Analyzer

The Analyzer provides basic statistical plots. It can be used to read the results of a simulation. However, we developed our own tool (Chapter 6 for more details) which provide additionnal features such as exportation into **.PNG, *.CSV* files and the most important for our work : average values from thousands of files.

3.5 Programming interface

The programming interface provided by QualNet is defined in several source files and mainly depend on the protocol you want to modify/add. Each layer(*Antenna,Physical,Mac,Network,...*) has its own generic source file which dispatch calls from upper/lower layer to the right protocol functions. In order to add a new layer or antenna model, it is necessary to modify each method defined in the layer generic source file. For example, if one wants to add a new MAC layer model, all the methods in the file `mac.cpp` must be modified to take the new MAC into account. For full integration, it is also required to implement in the new source file each method defined in the generic source file.

Chapter 4

Implementation

4.1 Code structure

This chapter describes the implementation of the AAA (ULA and UCA geometries) in the Antenna layer of QualNet. Our implementation of AAA is written in *C++*, a third party Template Matrix library[1] is used and Kei Takayama (TUAT) provided us an *Additive White Gaussian Noise* library to generate the noise in the system.
There are two main classes : *SourceofSignal* and *AAA*.
SourceofSignal (abbreviated *SoS*) represents a signal, its parameters are :

Direction of Arrival (int) The DoA of the signal as an azimuth, expressed in degrees (0 = north, clockwise QualNet convention), switched off under Rayleigh fading since the DoAs of the rays are choosed randomly.

Power (float) Reception power of the signal at the antenna, in Watt.

Modulation Type (int) BPSK (0) and QPSK (1) are supported

Length (int) Number of bits that the signal carries

The AAA class is used to represent the AAA, the parameters are :

Element count (int) The number of antenna elements

Type (int) The shape of the antenna, either Array (*ULA*)(value = 0) or Circular (*UCA*) (value = 1)

Element spacing (float) The space between elements in the case of a ULA | the radius in the case of a UCA. This value is expressed in fractions of the wavelength

[1] http://www.techsoftpl.com/matrix/

Frequency (float) The carrier frequency at which the AAA operates, used to calculate the wavelength ($\lambda = C/f$) and the size of the antenna. Set to 1 Hz by default.

Ray count (int) Number of incoming Rayleigh faded rays (MAX 20 | 0 or 1 = no fading)

SNR (float) Input SNR of each element, used to generate the noise

When a signal is created, it generates its own data. Once signals and AAA are created, there are three steps to calculate the Wiener solution:

1. Signals (desired and interference) must be added at the receive antenna by calls to the function *void AAA::AddSource(SoS*)*, which generates fading rays' DoAs, phase shifts and calculates the *steering vector*($\mathbf{a}(\theta)$ in equation 2.13)

2. The reference signal is set by calling *void AAA::setReferenceSignal(SoS*)*, the *SourceofSignal* must be the one transmitting the desired signal. The AAA will extract data from this *SourceofSignal* to construct the reference vector (r in equation 2.17)

3. A call to the methode *float AAA::ComputeWeight()*, will result in the computation of the Wiener solution by constructing and filling matrices with signals' input and noise, and by calculating the optimal weight vector ($\mathbf{w}$ in equation 2.17)

4. Call *AAA::getSignalGain(SoS*)*, this function will calculate the antenna gain for this signal with the method described in Section 2.7.

The *weight vector* can be retrieved by calling the method *Matrix AAA::getWeight()*. It is used by the AAA model in QualNet to store the computed pattern

4.2 First attempt

After coding the AAA implementation in Eclipse for Kamiya Sensei's exercises, we included our code in the Physical 802.11 layer to obtain the signal's DoA and the reception power. In order to fully implement the AAA into the PHY layer, we must modify many functions and we do not have a generic layer which could be reused with other PHY layers. Therefore we decided to create a brand new antenna model.

4.3 AAA model

QualNet already provides some Antenna models : omnidirectional, steerable and patterned. We based our modifications on the patterned source code.

Implementing AAA in the antenna model is easier than into the PHY layer. Unfortunately QualNet calls the PHY layer to notify about an incoming signal and the PHY layer queries the antenna about the gain for the azimuth of the incoming signal, so the antenna cannot directly know if the signal is a desired one or not. Antenna Model has two functions that the PHY will call to *optimize* the directional gain : *Antenna_MaxGainPatternForThisSignal* (PHY layer wants to maximize the gain for this signal, a desired one) and *Antenna_SetBestPatternForAzimuth* (PHY layer wants to maximize the gain in a direction which is the incoming direction of a desired signal). When these functions are called, we can calculate the Wiener solution for a desired signal and we then *lock* the antenna, which will be *unlocked* when the PHY layer changes the pattern back to the *default* (omnidirectional) pattern. The antenna will also be locked when the PHY layer selects another pattern (different from the default one). When the PHY layer queries the antenna for the gain in a direction, the AAA checks if it is locked and if the direction is different from the one on which the antenna locked itself (desired signal DoA). If these conditions are verified, the gain is for an interfering signal. The AAA computes a new Wiener solution with the desired signal DoA and the new interfering signal and then returns the gain for the interfering direction.

4.4 Simplifications

As the QualNet simulator does not compute every detail (de/modulation) of the simulation, the AAA implementation does not accomplish all the work a real AAA would have to do. For example, when a interfering signal arrives during the reception of a desired signal, the AAA computes the Wiener solution with two signals starting at the same time, ignoring the starting time of the two signals, which can be different.

4.5 Potential optimizations

Since the Wiener solution is very computing intensive, we decided to store computed patterns (i.e., the weight vector) to reduce the simulation time. Each time a new desired signal is received, the antenna checks in a table if the signal's DoA is already known. If it is not, a new Wiener solution is computed and the new weight vector and store it, along the DoA(s) that produced it, for future use. But this pattern storage is not suitable for mobile scenarii as the DoA between nodes changes over time. Since our goal is mobile scenario and because of the code complexity induced by the pattern storage, we removed it and rewrote the source code for readability and computation speed improvement.

4.6 Fading integration

QualNet engine does not provide the DoA of the Rayleigh fading rays. It computes an attenuation in dB instead, before notifying the PHY layer of a reception. This attenuation, correct for omnidirectional antenna, does not help us to compute the attenuation with an AAA, so another Rayleigh fading and Doppler effect (due to mobility), were implemented in our work. Fading level is expressed by the number of incoming rays; 1 ray is similar to the *Line of Sight* (no fading), the maximum (but can be changed in source code) is 20 rays. See Chapter 5 for Adaptive antenna Gain and SNR values under fading.

4.7 Statistics

QualNet provides several statistical results after a simulation, these statistics depend on the layer. To have a better view of the network behavior, some statistics have been added to the PHY layer and others created in the AAA Antenna (QualNet developers did not implement statistics in other antenna model) :

4.7.1 PHY 802.11

The statistics have been added to get more useful values, percentages are easier to compare than number of received or erroneous signals. We list here the statistics added :

Percentage of erroneous signal (%) QualNet only provides the number of erroneous signal (considered as corrupted by QualNet on a BER estimation), it is now divided by the number of received signals, so the value is more understandable and facilitates the comparison of different simulations

Percentage of interfered signal (%) the percentage of interfered signals on received signals

Percentage of error in interfered (%) the percentage of error in desired signals under interference only, useful to compare performances between omnidirectional antennas and AAA, which are more efficient for extracting a desired signal from interference and noise.

Average interfering signals at a moment (scalar) during a reception, other incoming signals are counted and an average is computed.

4.7.2 AAA model

The following statistics are used to analyze the behavior and the performance of the AAA. Additionnal statistics are visible in Figure 6.1 in Chapter 6.

Antenna Lock Count (scalar) the number of times the antenna has been locked on a signal

Worst Gain (dB) the worst gain the antenna returned to the PHY layer, usually for an interfering signal

Best Gain (dB) the best gain returned by the antenna for a desired signal

Gains Asked Count (scalar) the number of calls from the PHY layer

4.8 Parameters

QualNet configuration files list the parameters used by layers for their configuration. The new AAA antenna has its own list of parameters (parameters from Section 4.1). If a parameter is not present in the configuration file, a default value is used :

ANTENNA-AAA-ELEMENT (int) the number of antenna elements (default 4)

ANTENNA-AAA-TYPE (int) 0 = Linear | 1 = Circular (default Linear)

ANTENNA-AAA-ELEMENT-SPACING (float) Size between each element in the case of a Linear or the radius in the case of a Circular, it is expressed in terms of λ (Wavelength) which depend on the frequency of the carrier (d in Figure 2.12) (default 0.3)

ANTENNA-AAA-SNR (float) Input SNR at each element, used to generate the *Additive White Gaussian Noise* (default 10.0)

ANTENNA-AAA-RAY (int) Number of Rayleigh fading incoming rays (MAX 20 | default 0)

Chapter 5

AAA Implementation Validation

This chapter describes the Gain and output SNR values of our implementation of the AAA (ULA and UCA geometries) in the Antenna Layer of QualNet. ULA and UCA were tested with a number of elements ranging from 1 (corresponding to an omnidirectional) to 10. We consider three different scenarios of fading : absence (**-LoS*), 5 and 10 Rayleigh fading rays (**-Fading*). We report on charts the antenna gain and output SNR of an interference-free signal (*no Interference-**), of an interfered signal (*desired-**) and of its interference (*interferent-**).

The antenna parameters are :

- $AAA_{ELEMENT-SPACING} = 0.25$
- $AAA_{SNR} = 10dB$
- 50 bits long message, QPSK modulation
- $Power_{Signal} = 1.78\mu W$
- 2,000 iterations to compute average values
- $AAA_{Elements} = 1\ through\ 10$
- $AAA_{Ray} = 0; 5; 10$
- $AAA_{Type} = Linear|Circular$
- One interferent signal with the same power as the desired one.

5.1 Gain

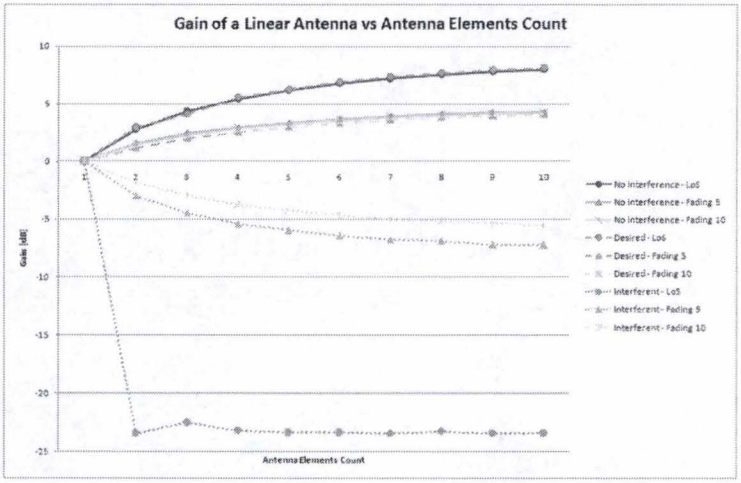

Figure 5.1: AAA Gain of a ULA vs Elements Count

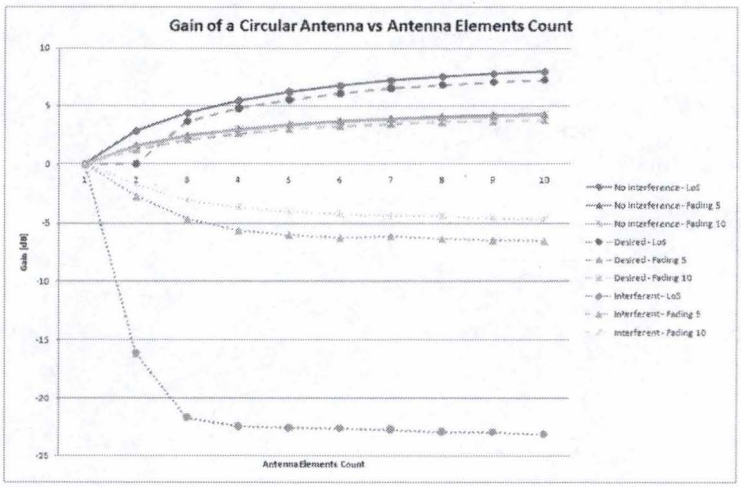

Figure 5.2: AAA Gain of a UCA vs Elements Count

As expected, the antenna gain for a *non-interfered - LoS* signal is near the theoretical optimal gain of an AAA which is $10 * log_{10}(N)$ where N is the number of elements. A *desired-LoS* signal follows the same curve, proving that an AAA can extract a desired signal data from interference

and noise efficiently. ULAs seem to be more efficient than UCAs, due to the redundancy of information since the circular shape is symmetric. Under Rayleigh fading, signals suffer from attenuation but *desired-faded* signals are still well received as *non-interfered-faded* signals, so the AAAs can handle interference even under fading.

5.2 Output SNR

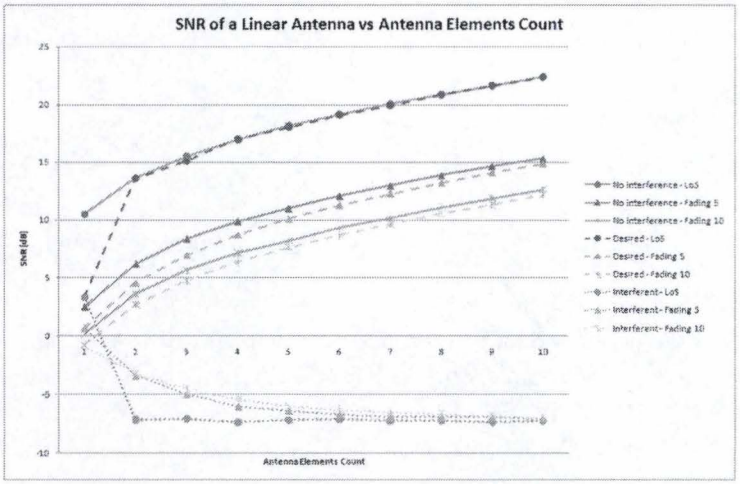

Figure 5.3: AAA output SNR for Linear Antenna vs Elements Count

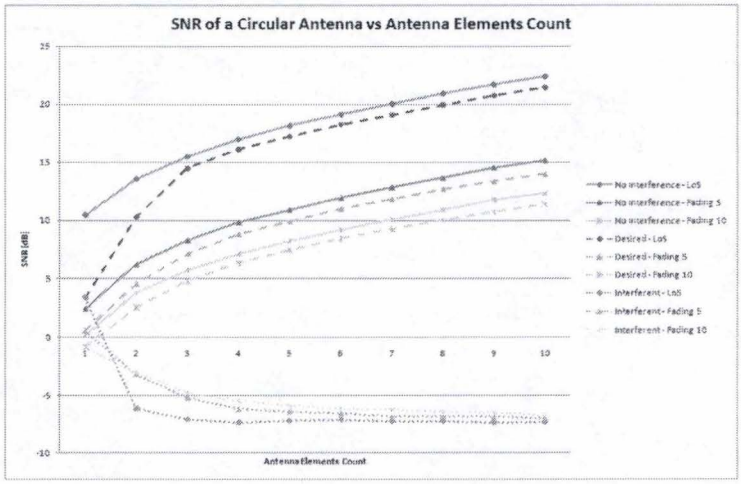

Figure 5.4: AAA output SNR for Circular Antenna vs Elements Count

Here, we can see a small difference between *non-interfered-faded* signals and *interfered-faded* signals, this difference decreases with a higher number of elements in the ULA. We also notice the difference between faded signals with 5 and 10 rays, the curves are separated by 3 *dB*.

5.3 Doppler effect

As mobility induces phase-shifts due to the Doppler effect, we simulated the same configurations as the ones described above with the Doppler effect for random values of movement speed and angle. Gain and SNR values given by the simulations are identical to the values obtained without the Doppler effect. It is understandable because the phase-shift is the same for every antenna element, so the AAA filtering suppresses completely the impact of the Doppler effect as long as Wiener solution derived within coherence time of channel.

Chapter 6

Java Tools

The QualNet GUI already provides tools to analyze statistics and to prepare batch experiment runs, but they do not work as we expected, so we developed small graphical tools in Java to meet our needs.

6.1 QualNetStat

With this tool, we can gather statistical results from QualNet's simulations (*.stat files) and compare many simulations on the same chart or export results to graphic files (*.PNG) or to *Comma Separated Values* (*.CSV) files in order to use them in spreadsheet softwares such as OpenOffice Calc or Microsoft Excel. The tool can also compute average values from multiple statistics files made from the same configuration (using different random seeds) and export them to csv. With a parameterized larger Heap value, we were able to load up to 2,500 result files at once to calculate average values.

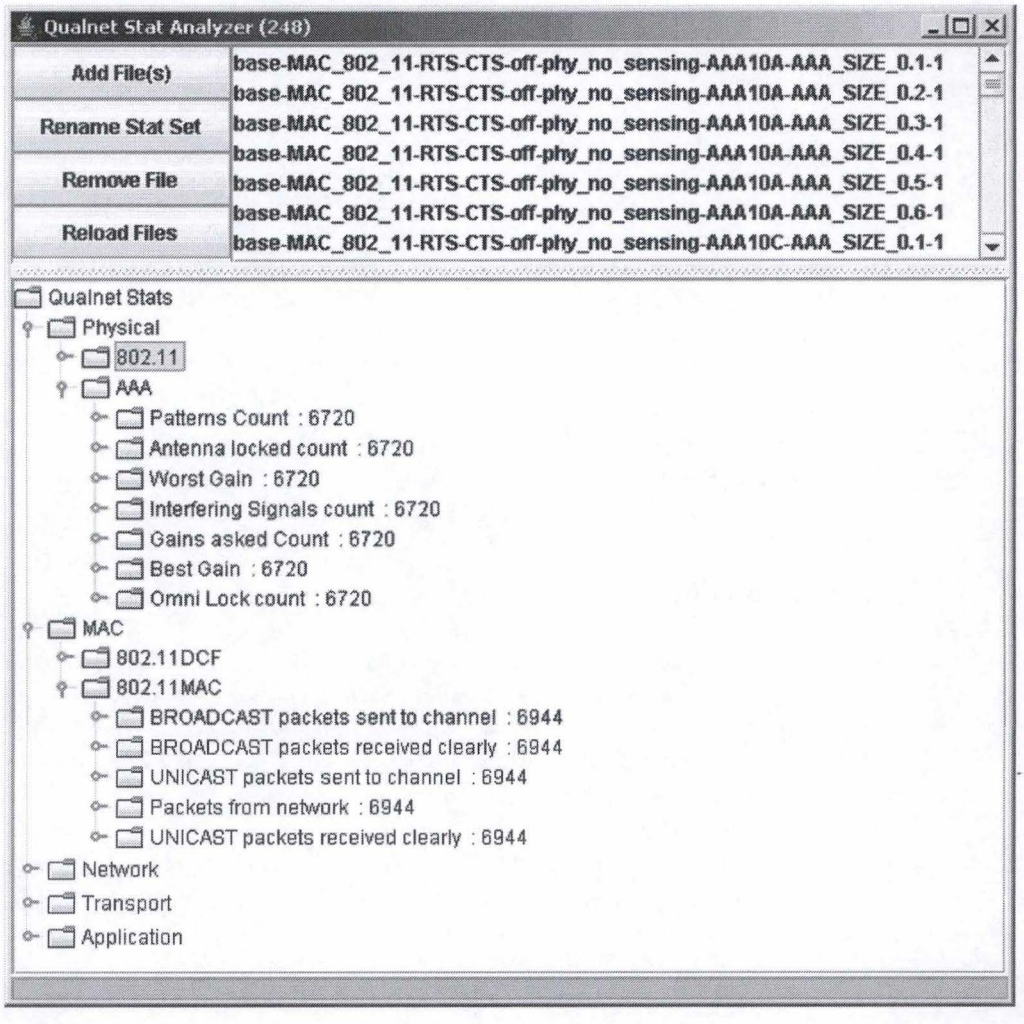

Figure 6.1: QualNet Stat Analyzer

Figure 6.1 is the main windows of the tool where we see in the upper-left corner, a group of buttons to load and manage statistic files. The list of loaded files is in the upper-right corner. The tree represents the hierarchy of protocol layers and under each protocol are listed the statistics provided by the layer, these statistic nodes store the average value for gathered files (same name but different seeds). Selecting a node pops up a menu to choose the different actions or to export in available formats for the concerned node.

6.2 QualNetBatch

A new antenna model, like AAA, needs a lot of simulations to compare behavior of *MANET* in different configurations. It is possible to prepare batch experiments in QualNet's GUI, but this embedded tool does not handle the subnet configuration (specific configuration for a set of nodes) which is mandatory in our experiment configurations and uses the graphical simulation execution which slows down execution speed. We developed a graphical tool to handle different parts of configuration files, to build a tree which contains every combination to be simulated by QualNet engine. The number of different seeds can also be set. The tool captures the output of simulations with filters in order to export them to files (AODV events for example) or to show the progress of simulations on the screen. An inactivity detection was also added, so when the license server handshake fails, it aborts the simulation and launches it again directly.

To further optimize the simulation execution time, we developed a tool similar to this one but which dispatches simulations to many computers across the network, see Section 6.4.

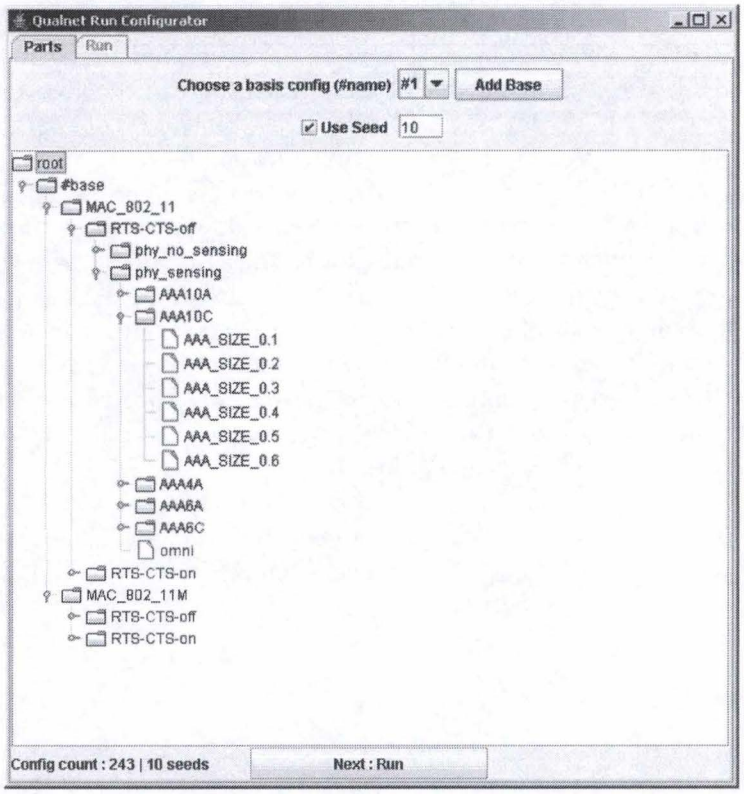

Figure 6.2: QualNet Batch configuration interface

Figure 6.2 shows the window to configure a batch run. The tree represents every combination of configurations that will be executed, the number of different seeds can be specified in the upper part of the window.

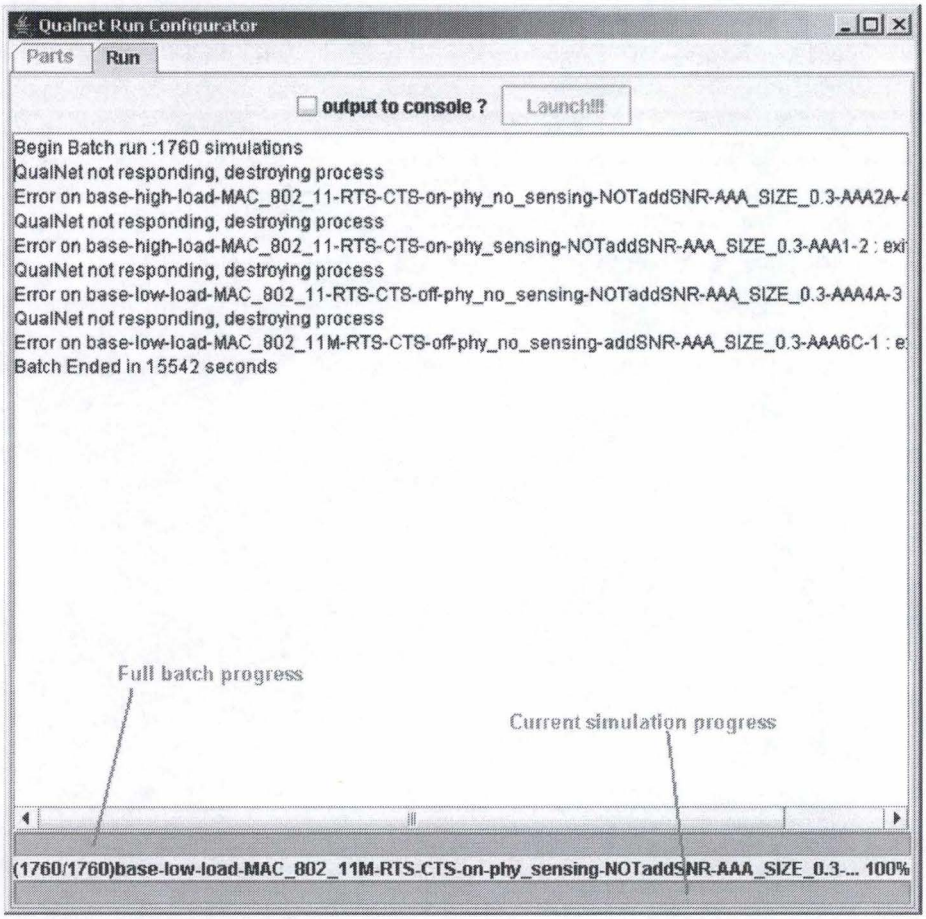

Figure 6.3: QualNet Batch Run Finished after 4 restarts due to License Server handshake failures

Figure 6.3 shows the batch execution process, error messages are always displayed and regular output can be switched off for clarity. We can see that some executions failed due to an error with the license server handshake. The tool handles this by killing a blocked simulation and launching it again. The two progress bars in the lower part show the progression of the current simulation (lower bar) and for the full batch (upper bar), The name of the current simulation is also printed on the screen.

6.3 QualNetAODV

When we attempted to analyze the behavior of AODV using QualNet, we noticed that the statistics provided by the QualNet AODV Network Layer are not very useful, as the only information provided for the hop count is the Total Hop Count for all routes for each node. We add the average length of all routes to facilitates the comparison between differents scenarios. We also developped a tool to obtain a graphical representation of the topology of the network through time. This tool uses the *.nodes* file of QualNet to obtain the location of the nodes. We have modified the AODV implementation provided by QualNet so that it outputs the changes of the routing table for each node. The QualNetBatch tool generates an *.aodv* file from these outputs.

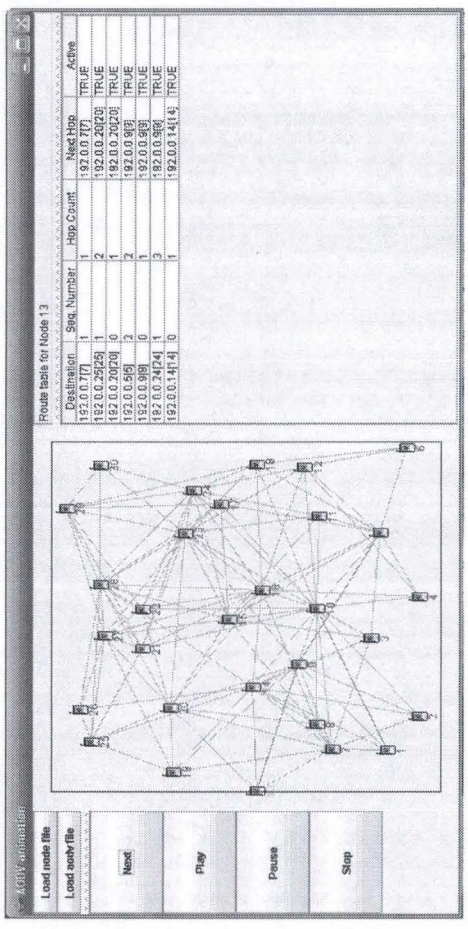

Figure 6.4: AODV Viewer main frame

At first, the *.nodes* file must be loaded by using the top button. Afterwards, the *.aodv* file must be loaded by using the second button. The *next* button is used to advance time by one second (many steps could happen in this time). It is possible to see the current routing table by clicking with the middle mouse button on a node. The left mouse button is used to select a source for a route to lookup while the right button is used to select the destination. The green links represent the active links, the red ones show the inactive links. The route is highlighted in purple. Nodes mobility is not yet supported.

6.4 QualNet Dispatching Service

As we have many simulations to execute, a new tool is developed to enable us to dispatch simulations on many computers across network (or internet). This tool is also written in Java and use object serialization to send messages, tasks and files across the network. The application is structured in three main components : client interface, dispatching server and simulation node. The server receives connections from client interfaces and simulation nodes; the user prepares simulations batches in the client interface, submit them to the server with linked files (**.nodes, *.app, *.config, ...*). The server dispatches batches to almost idle simulation nodes, which rebuild configuration files, run simulations, gather statistics and send them to the server. The user can follow the progression of work in real time through the client interface.

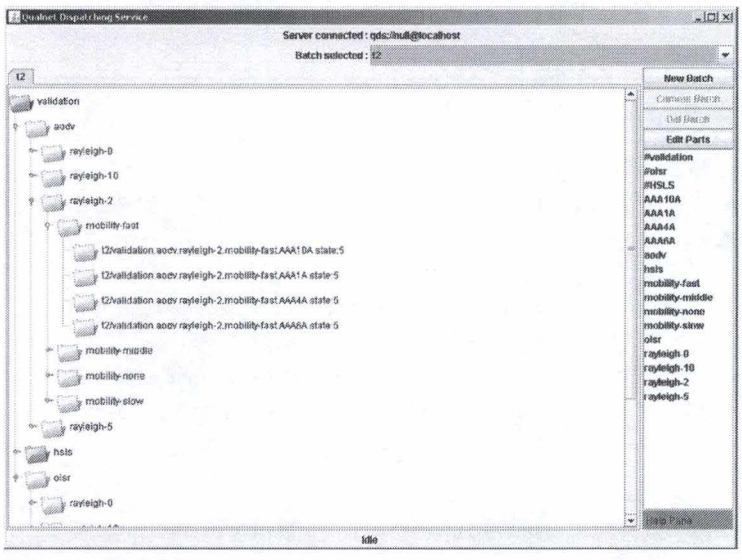

Figure 6.5: QualNet Dispatching Service Client Interface

Chapter 7

Modifying MAC and PHY layers for AAA

As shown in Chapter 5, AAA performs equally well under interference or without interfering signal, so *space reusability* seems possible with an AAA. Due to CSMA/CA of the 802.11 MAC layer, a node does not transmit if the medium is busy with another signal. The ability of AAA to handle interference paves the way for *Space Reusability*, that is to say the ability for concurrent transmissions to occur without collision as they are spatially separated.

To improve network efficiency thanks to *Space Reusability*, we need to modify some medium access mechanisms without completely designing a new MAC layer.
Two modifications concern the MAC 802.11 layer, and one the PHY layer. The first modification is to change the MAC layer in order to ignore signals which are not destined to the node (see Section 7.2). This modification combined with the deactivation of the *sensing* state of the PHY layer (see Section 7.1), enables the node to transmit a frame directly even if the medium is busy with a transmission between two other nodes.

Another step was to suppress the *RTS-CTS* handshake (see Section 7.3), which proved to be useless in some researches [ ].
Here, we explain which modifications have been performed to the original QualNet source code in order to achieve these optimizations. We will conclude with simulation results.

7.1 Disabling sensing in PHY layer

The 802.11 PHY layer in QualNet uses the function *Phy802_11CarrierSensing* to return *true* in case of a busy medium. We added a new parameter

PHY802.11-NO-SENSING which sets a boolean flag used in the function to return *false*, so the medium is considered to be idle even if it is not, but interference power is not reduced.

7.2 802.11 MAC ignoring unicast messages to others

The 802.11 MAC layer has a function *Mac802_11ExaminePotentialIncomingMessage* which calculates the time needed to receive the header, changes the state of the layer (*idle, waiting CTS, waiting ACK, waiting header*) and starts a timer with the computed time. Once the time expires, meaning that the header has been received, the timeout handle function (called after a timer) performs actions depending on the state of the layer. In this case (*waiting header*), it checks the destination of the incoming packet, the probability of an error (BER based) and decides whether to stop the reception or not. If the reception has been stopped, it checks for another outgoing packet with no delay. This behavior was implemented by QualNet developers but not thoroughly tested. Some bugs remained, which we corrected. We also modified the waiting delay which was originally set to the end of a full packet reception. We removed the waiting time in order to achieve *space reusability*. In the original code, QualNet processes the header to eventually trigger a backoff delay if a RTS or CTS is caught. We suppressed this processing as RTS-CTS are messages to another node to reserve the medium. This reservation is useless with the *Space Reusability*.

7.3 MAC without RTS-CTS

In the 802.11 MAC layer, the function *transmitFrame* decides whether to send a RTS or not depending on the type of destination address (unicast or broadcast). In the case of a unicast, we added a conditional switch : if the *MAC-802.11-RTS-CTS-MODE* parameter is set to *NO*, it transmits directly the data frame with no RTS-CTS handshake. Otherwise, if the parameter is set to *YES* (default value) it sends a RTS frame. Unicast data packets still trigger an ACK frame from the receiver.

7.4 Parameters

Here is the list of parameters used to modify the behavior of the 802.11 PHY and MAC Layers:

PHY802.11-NO-SENSING YES = disable the sensing state | NO = default behavior

MAC-802.11-STOP-RECEIVING-AFTER-HEADER-MODE YES = MAC stops reception in case of unicast to another node | NO = default behavior

MAC-802.11-RTS-CTS-MODE NO = do not send a RTS frame before sending the data frame | YES = default behavior

Chapter 8

Static scenarios & Results

Two different networks are simulated :

Parameters	**Test A**	**Test B**
Node count	28	100
Terrain size	1500m x 1500m	4000m x 4000m
Data flow count	14 UDP	8 UDP
Routing protocol	AODV	AODV
Mobility	None	None
Fading	None	None
Seeds used	10	3

Test A 14 connections, with 512 bytes packets, the delay probability before each transmission (inter-packet departure time) follows a uniform distribution between 10 and $50ms$. All communications start at the same time and last $25s$.
Each configuration was simulated 10 times with a different seed[1]. The values reported in the result charts are the average values of these 10 simulations.

Test B 8 connections with a transmission rate of $50KB/s$ during $35s$, starting at different times.
Each configuration was simulated 3 times with a different seed. The values reported in the result charts are the average values of these 3 simulations.

[1]for the pseudo-random number generator

Results show the behavior of the network for the 8 different combinations of MAC and PHY parameters (see Figure 8.1).

- MAC

  MAC S Standard 802.11 MAC

  MAC M Modified MAC, ignoring unicast to other nodes

- RTS-CTS

  RTS RTS-CTS handshake before sending data

  NO RTS No RTS-CTS handshake

- Sensing

  Sensing PHY layer default behavior

  No Sensing the sensing state is assimilated to the idle state

The configuration *MAC S + RTS + Sensing* is the standard behavior of a 802.11 compliant node.

The configurations with *MAC M + No Sensing* can send pending data frames directly after aborting the reception of frames destined to another node. This can improve the *space reusability*. This behavior is expected to be more efficient with AAA because the antenna can cancel interference.

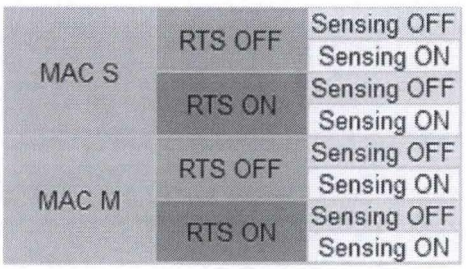

Figure 8.1: Parameters conbinations in result charts

8.1 Interference and transmission errors

8.1.1 Interference

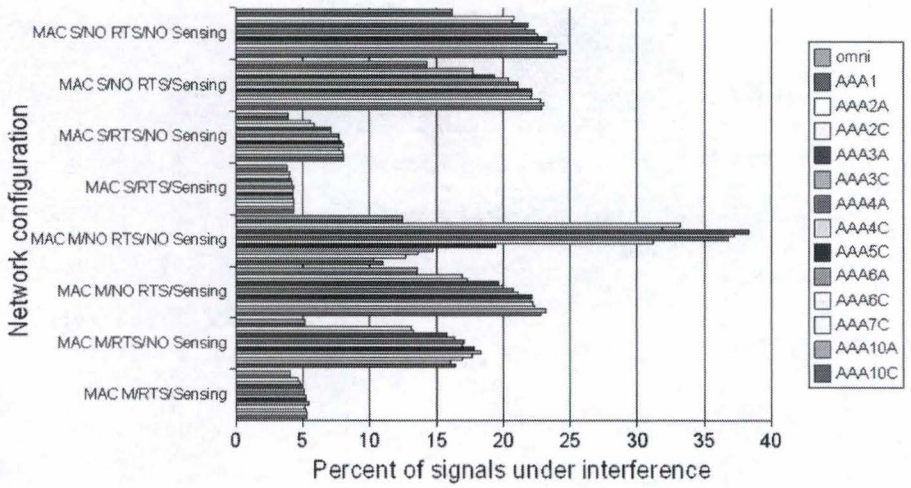

Figure 8.2: [Test A] Percent of interfered signals

Figure 8.2 shows the percentage of interfered signals in *Test A*. We clearly see that the RTS-CTS mechanism associated with the sensing state (configurations 4 and 8) avoid interference. On the other hand, *MAC M + No Sensing* (configurations 5 and 7) raises the level of interference.

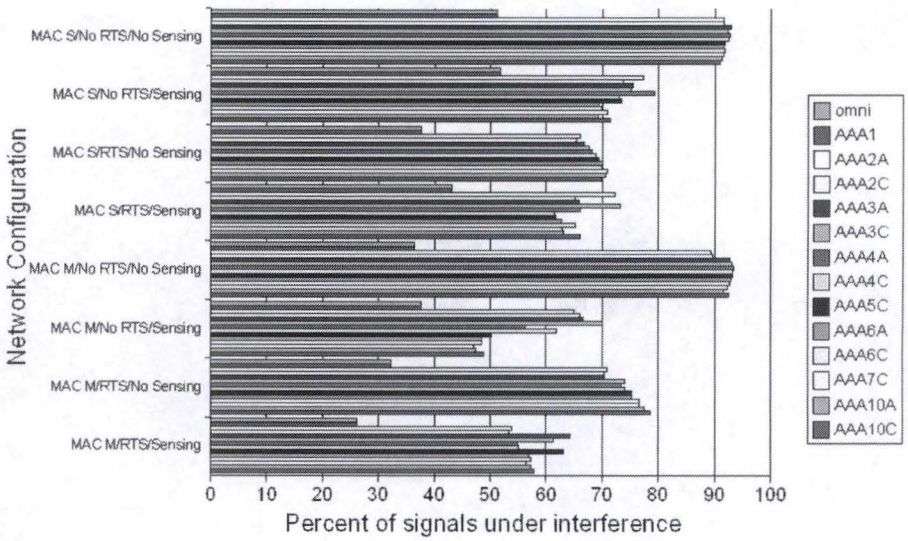

Figure 8.3: [Test B] Percent of interfered signals

Figure 8.3 shows the percentage of interfered signals in *Test B*. Due to the distances between the nodes, the levels of interference are much higher than in Test A, the scale of the charts are different. *No RTS-CTS and No Sensing* (configurations 1 and 5) cause the highest levels of interference, because there is no mechanism to control the access to the medium. Omnidirectionnal antenna (Omni and AAA1) cause less interference because of the backoff delays in the case of an ACK timeout (refer to the two following observations).

8.1.2 Transmission errors

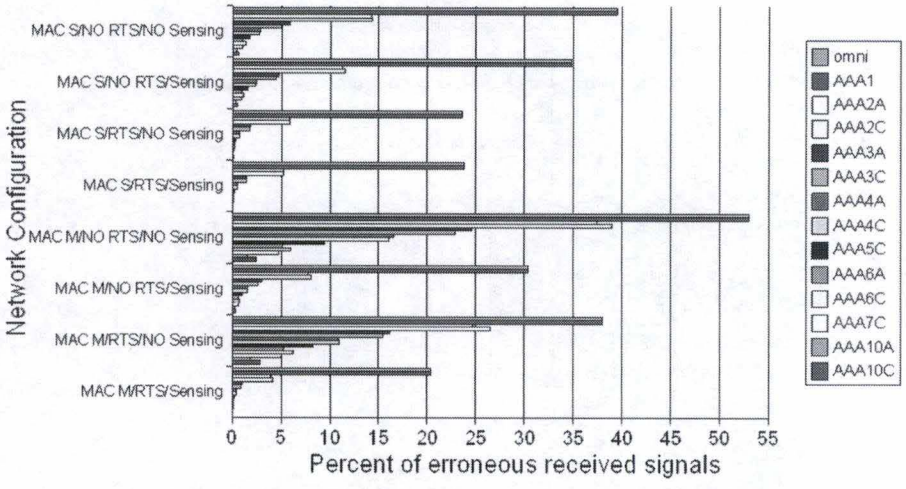

Figure 8.4: [Test A] Percent erroneous received signals

Figure 8.4 shows the percentage of erroneous received frames in *Test A*. In this test, the average percentage of errors depends mainly on the level of interference induced by the network configurations, because the nodes are close enough to communicate with small hop count. AAA with more elements have a lower error percentage than the ones with few elements. We will see the consequences in Application layer statistics.

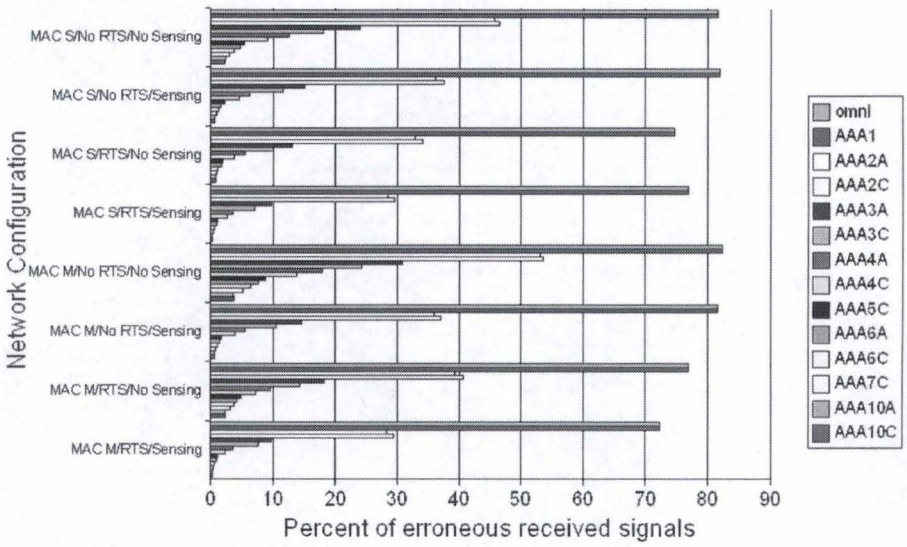

Figure 8.5: [Test B] Percent erroneous received signals

Figure 8.5 shows the percentage of errors in received frames in *Test B*. Because of the greater distance between the nodes, the percentages of errors are higher and seem to be less sensitive to network configurations. AAA with more elements are still more efficient.

8.1.3 Transmission errors in interfered signals

This statistic is a subset of the previous one. Here, we only consider erroneous received frames when the desired signal is interfered by another signal(s) on the medium at the same time.

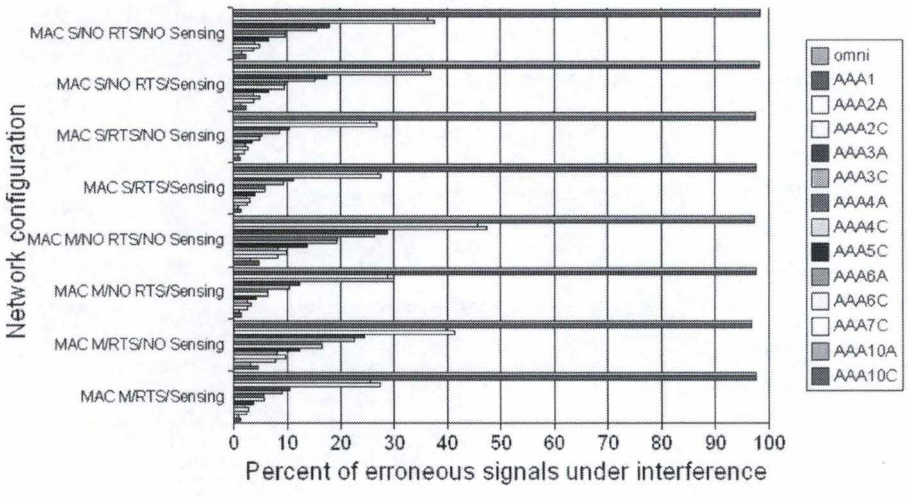

Figure 8.6: [Test A] Percent erroneous interfered signals

Figure 8.6 shows the percentage of errors in interfered signals in *Test A*. We can see that the efficiency of the AAA depends mainly on the number of elements. There is no significant difference between ULA and UCA. The network configurations which cause more interference, also cause more error in interfered signals.

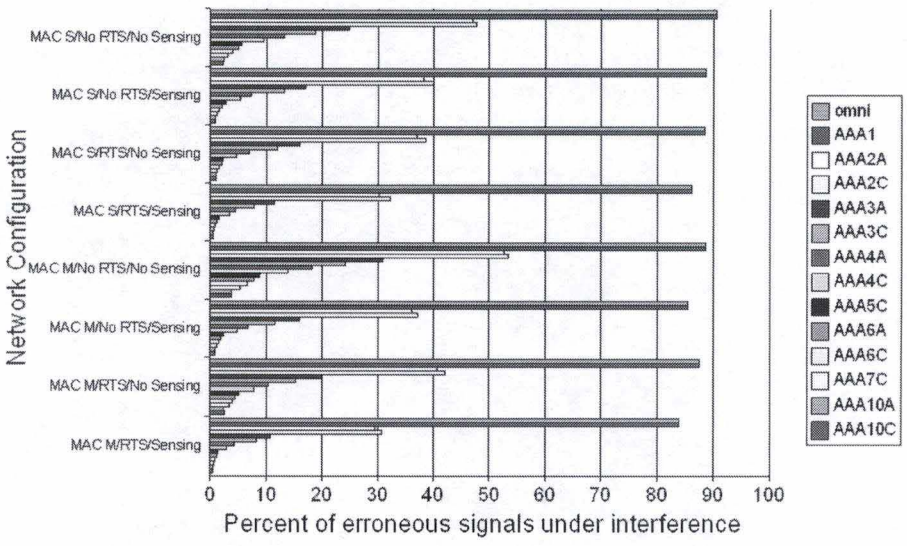

Figure 8.7: [Test B] Percent erroneous interfered signals

Figure 8.7 shows the percentage of errors in interfered signals in *Test B*. Again, the efficiency of the AAA depends mainly on the number of elements. The percentage of errors has almost the same value as in *Test A* except for omnidirectionnal antenna because of the greater distance between the nodes.

8.2 Application layer

8.2.1 Average end-to-end delay

The average end-to-end delay is computed only with the received frames.

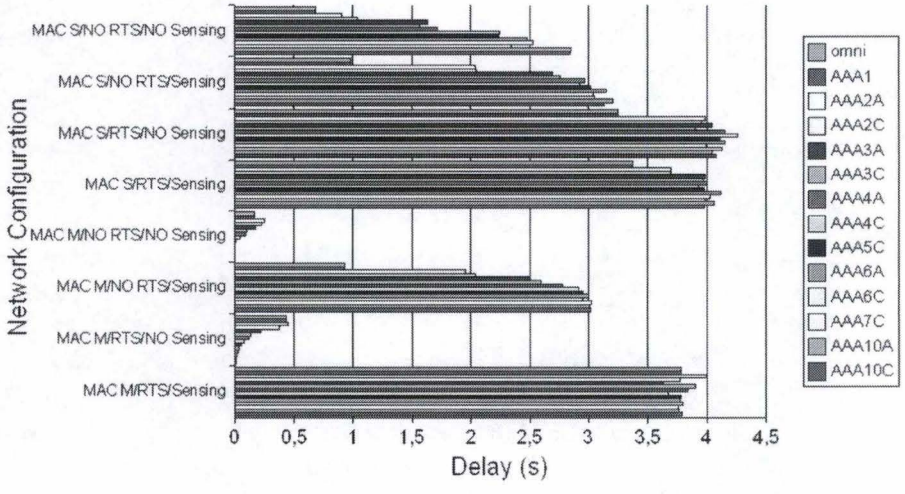

Figure 8.8: [Test A] Application layer : Average End-to-End delay

Figure 8.8 shows the average end-to-end delay in *Test A*. The delays are high because, in Test A, we simulate a great number of communications in a small and dense network. *MAC M + No Sensing* configurations achieve best delay, because the nodes do not wait to send data frames. Even with *MAC M*, disabling RTS-CTS handshake reduces delay.

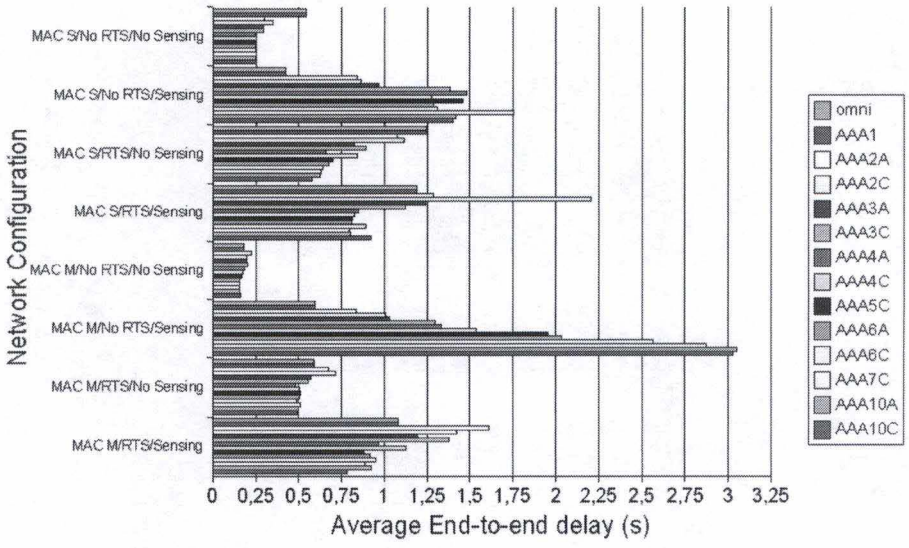

Figure 8.9: [Test B] Application layer : Average End-to-End delay

Figure 8.9 shows the average end-to-end delay in *Test B*. The delay values are lower due to a reduced traffic in a wider network, so the network is less congested. Disabling the sensing reduces delays specially with *NO RTS*. There is a weird behavior in network configuration 6 where the delay increase with the number of elements. The following observations about *bytes received* will explain this behavior.

8.2.2 Bytes received

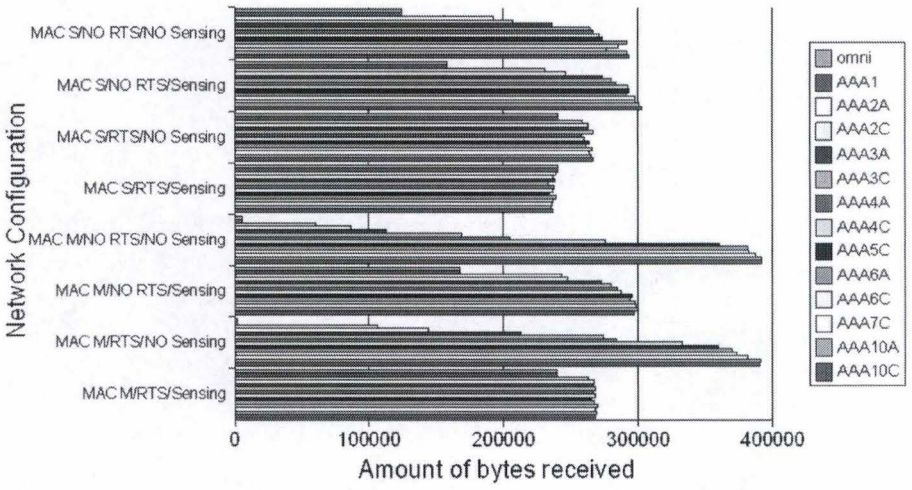

Figure 8.10: [Test A] Application layer : bytes received

Figure 8.10 shows the average number of bytes received at the end of the simulations in *Test A*. The average number of bytes received for every network configurations is nearly the same. The network configurations 5 and 7 (*MAC M + No Sensing*) present results which depend mainly on the number of elements. Omnidirectional antenna could transmit just a few bytes due to high interference levels (this small amount of bytes caused a miscalculation in the delay and throughput values).

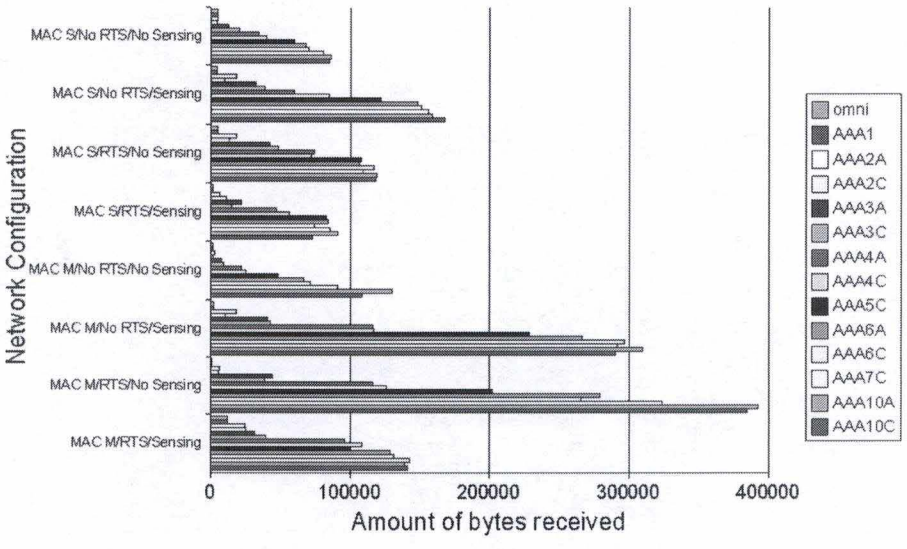

Figure 8.11: [Test B] Application layer : bytes received

Figure 8.11 shows the average number of bytes received at the end of the simulations in *Test B*. These values are not comparable with the ones in Test A, because the emission rates are different. In this test, configuration 6 and 7 achieve better results than other configurations but these results depend mainly on the number of elements. The remark about omnidirectionnal antennas in the previous observations still applies.

8.2.3 Throughput

QualNet computes throughput by dividing the amount of received bytes by the reception time of this bytes. This causes miscalculation when only one frame of data is received, causing the throughput to be equal to its maximum value.

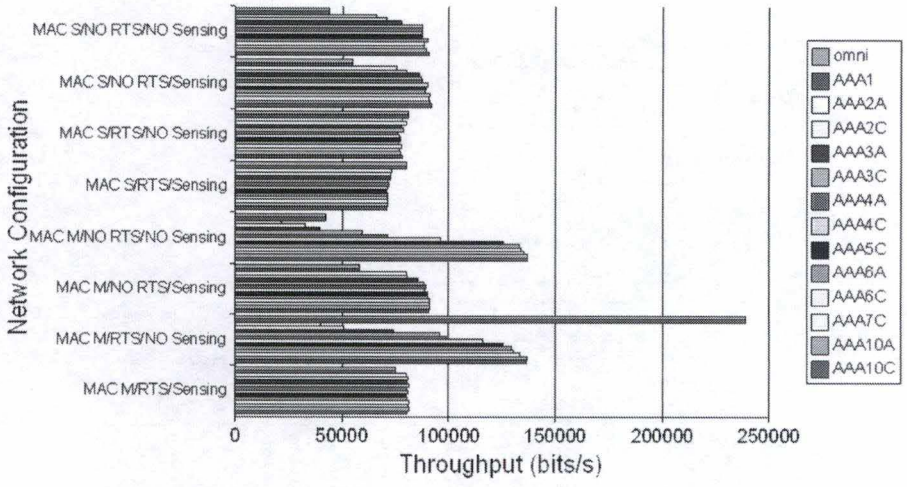

Figure 8.12: [Test A] Application layer : received throughput

Figure 8.12 shows the average throughput in all the simulations in the *Test A*. Like in the observations about amount of received bytes, network configuration 5 and 7 achieve better results than other configurations but still depend on the number of elements. The overvalued throughput of omnidirectionnal antenna in configuration 7 is due to the previously mentionned miscalculation.

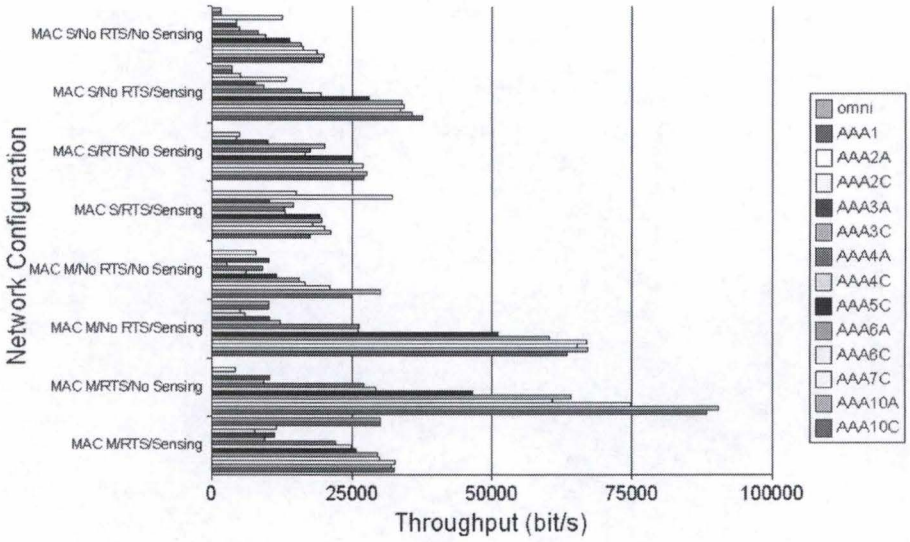

Figure 8.13: [Test B] Application layer : received throughput

Figure 8.13 shows the average throughput in all the simulations in the *Test B*. Configuration 6 and 7 yield better results than the others. The miscalculated throughput values of omnidirectional antenna in configuration 3, 4, 5 and 7 were removed because they caused the results chart to be unreadable.

8.3 Routing layer : AODV

8.3.1 Total hop count

This QualNet AODV layer statistic do not represent the *real* hop count in AODV routes, because each time a link is broken, AODV finds another route and add the new hop count to the hop count sum. So AODV routes seem to be too long for an emitting node which have only one destination during the entire simulation.

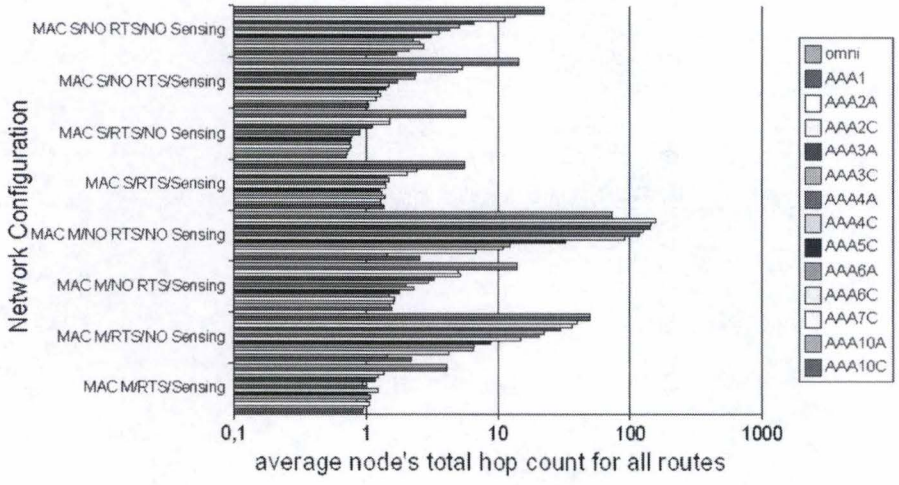

Figure 8.14: [Test A] AODV : average total hop count for all routes

Figure 8.14 shows the total hop count of all routes in *Test A* (notice the logarithmic scale). Here is the counter effect of *AAA optimized* configurations (5 and 7): They cause a greater hop count in AODV routing due to the high levels of interference which cause many link to break and frequent routes updates, so the sum of all routes is higher. In general, AAA with more elements yield better results.

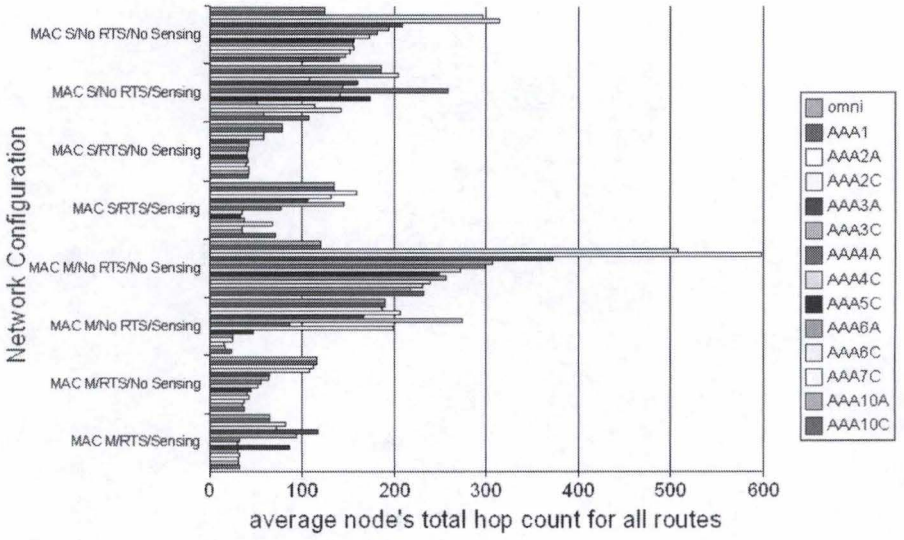

Figure 8.15: [Test B] AODV : average total hop count for all routes

Figure 8.15 shows the total hop count for all routes in *Test B*. In this test there are less broken links so routing is more efficient except in configuration 5. The configuration 7 which achieves good results in *bytes received* and *throughput* gives good results here as well.

8.3.2 Broken links

In the AODV protocol, when a communication between two nodes fails, AODV considers that the link is broken and tries to find a new route to the destination.

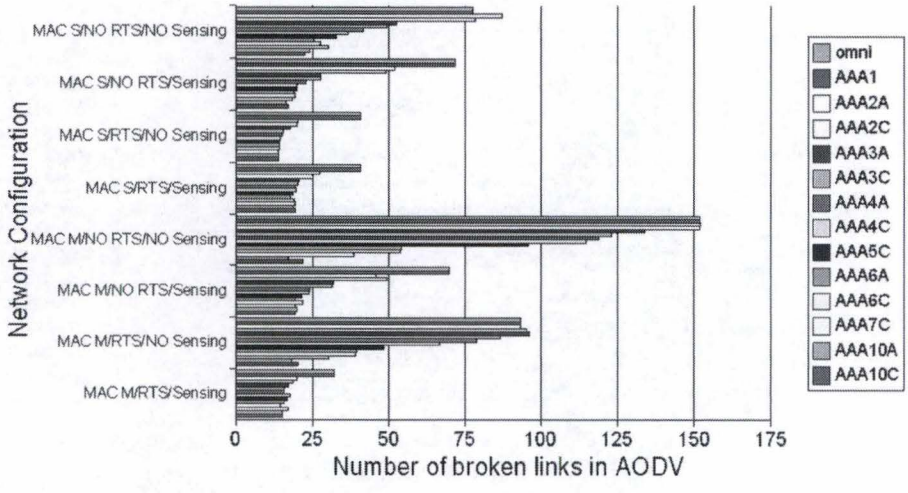

Figure 8.16: [Test A] AODV : average broken links

Figure 8.16 shows the average number of broken links in *Test A*. Network configuration 5 and 7, which cause much interference, suffer from many link breaks, but more elements improve the results since it handles interference much better.

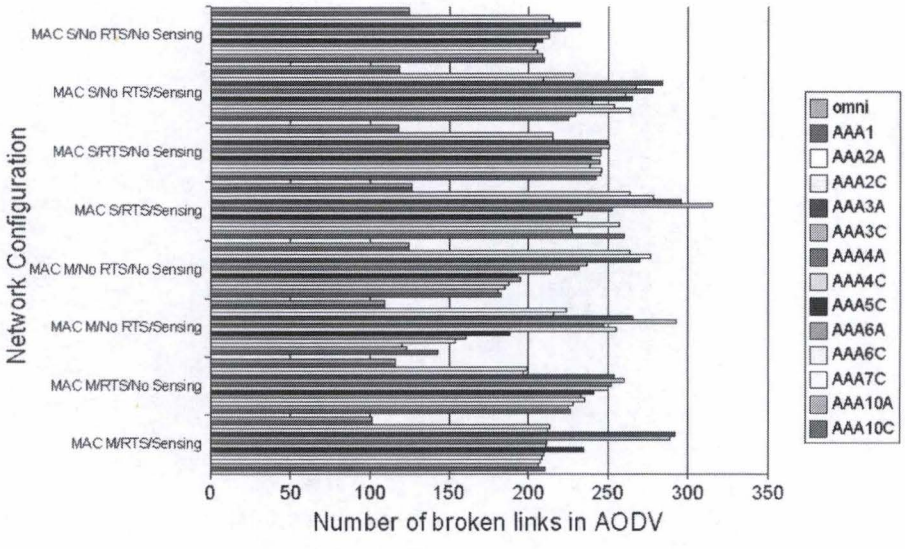

Figure 8.17: [Test B] AODV : average broken links

Figure 8.17 shows the average number of broken links in *Test B*. In this test, network layer configuration do not seem to influence the number of broken links, except in configuration 5 and 6 where arrays, with more elements behave better. Omnidirectional antennas seem to be more efficient than AAA, but they send few data frames and communications are slower.

8.4 MAC layer

8.4.1 Frames transmitted over the air

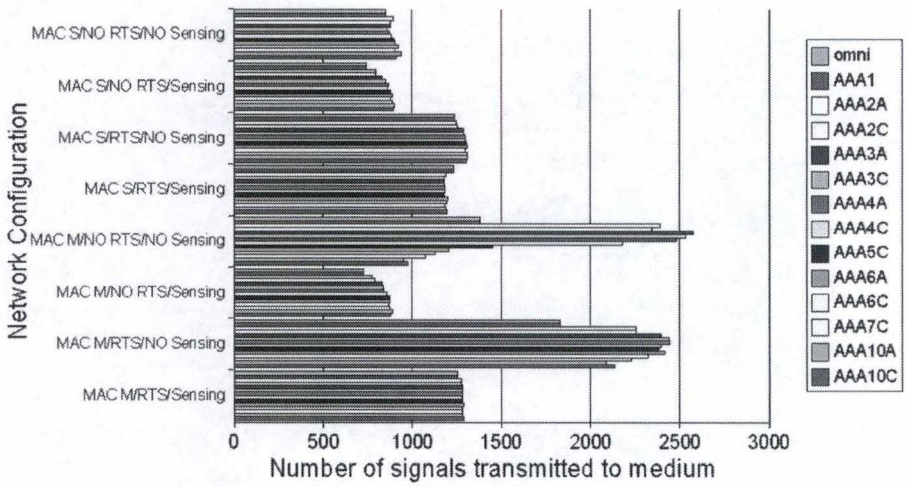

Figure 8.18: [Test A] Signals transmitted count

Figure 8.18 shows the average number of frames (data and control) sent by a node during a simulation of *Test A*. We can notice that disabling the RTS-CTS handshake reduces the number of frames sent. As expected, configurations *MAC M + No Sensing* (Config. 5 & 7) send more frames than other configurations because of a higher number of errors and consequently more retransmissions.

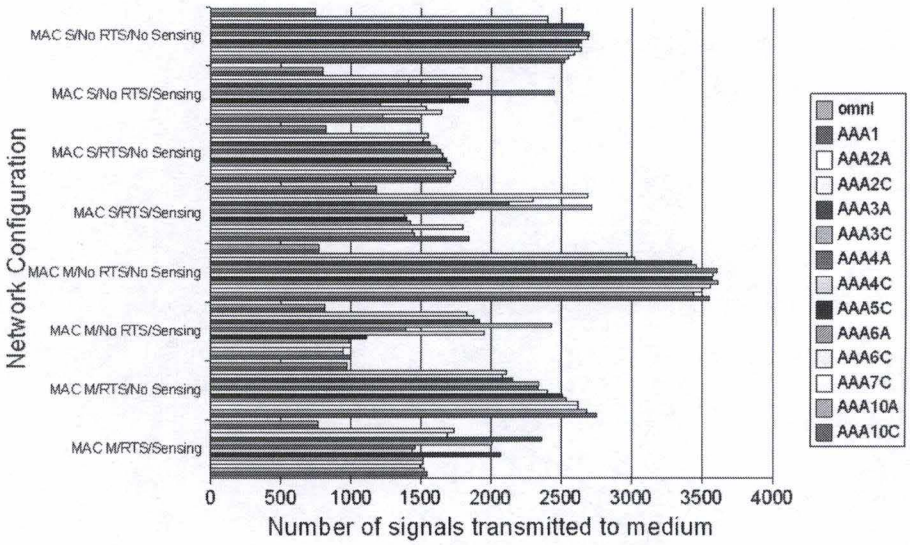

Figure 8.19: [Test B] Signals transmitted count

Figure 8.19 shows the same statistics as in the previous chart but for simulation of *Test B*. Values are less stable for a given network layer configuration (see Figure 8.1), but we are unable to explain this fact. Configurations 5 and 7 sent more frames due to higher interference levels. Surprisingly configuration 1 also sent more frames than the default 802.11 layers configuration.

8.4.2 Frames retransmitted over the air

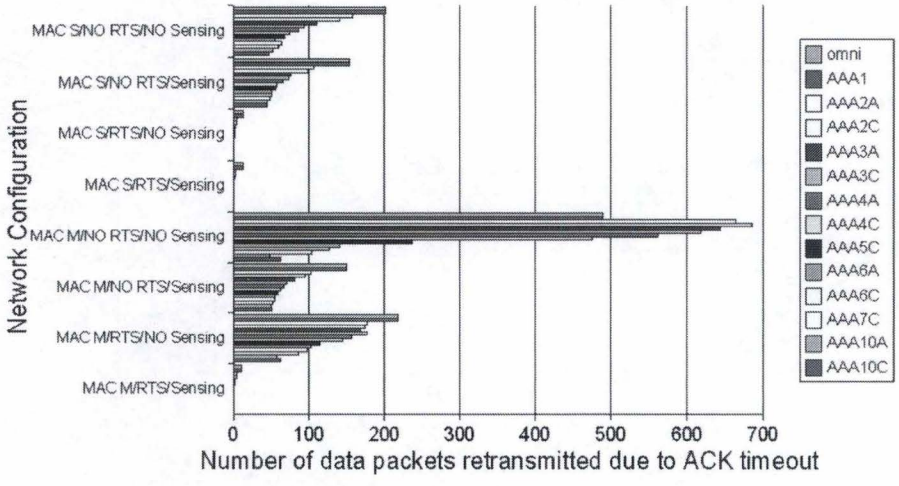

Figure 8.20: [Test A] Retransmissions due to ACK timeout

Figure 8.20 shows the number of retransmitted (unicast) frames in *Test A*. *RTS* configurations achieve better results than the ones without *RTS*, since they avoid interference and also reception errors. Configuration 5 triggers many retransmissions.

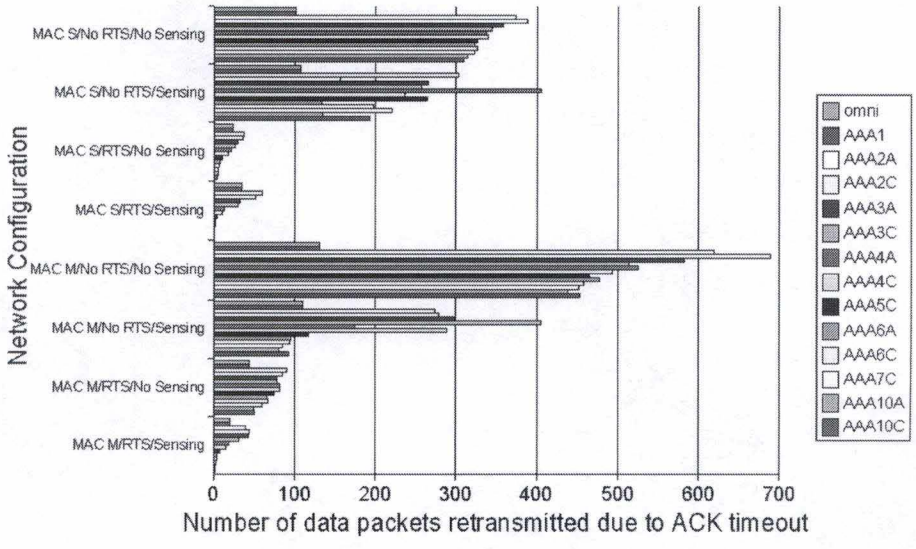

Figure 8.21: [Test B] Retransmissions due to ACK timeout

Figure 8.21 shows the number of retransmitted (unicast) frames in *Test B*. Again, *RTS* configurations also behave better than without the *medium reservation* mechanism.

8.4.3 Frames transmitted from PHY to MAC layer

The number of frame sent by the PHY layer to the MAC layer.

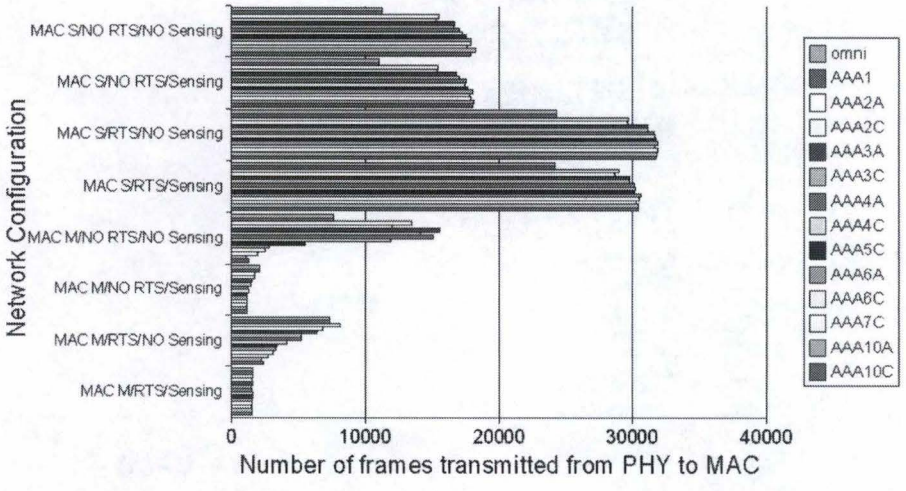

Figure 8.22: [Test A] Number of frames transmitted from PHY to MAC layer

In Figure 8.22, the MAC M configurations sent less frames to the MAC layer, because it ignores unicast frames to other nodes. In MAC S configurations, those with RTS-CTS handshake enabled, receive more frames because of the *RTS* and *CTS* frames.

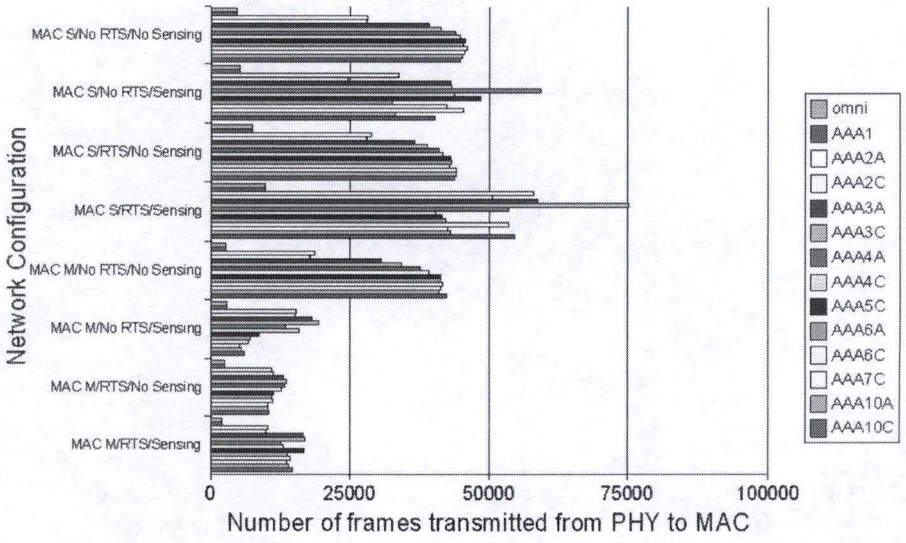

Figure 8.23: [Test B] Number of frames transmitted from PHY to MAC layer

In Figure 8.23, MAC M configurations sent less frames to MAC layer, but the RTS-CTS effect is not visible.

8.5 Conclusion

In an operating MANET, users only see the behavior of the Application Layer. When using AAA devices, the efficiency of the network mainly depends on the density of the network ($nodes/km^2$), the number of data flows and the throughput of these data flows.
From the different combinations of network layers modification, we see that with the classical network configuration (*MAC S + RTS + Sensing*) the network efficiency is not as bad as we expected it at the beginning of our simulations. Some configuration like *No RTS + No Sensing* can improve the throughput and delay but cause a lot of retransmissions, so the global efficiency of the network is lower. A new Directionnal Medium Acces Control (DMAC) could probably improve greatly this efficiency.
There is still a lot of simulations and work to do, to figure out an optimized network configuration specially designed for AAA equipped devices in a Mobile Ad-hoc Network.

Chapter 9

Mobile scenarios & Results

For these simulations, we create a mobility file in order to have the same node movements for every simulation. We set the parameters as follows:

Parameter	*Value*
Terrain size	1500m x 1500m
Antenna type	**omnidirectional, linear 4 and 10 elements**
Fading	**No fading, 5 rays, 10 rays**
Routing protocol	**AODV, OLSR**
Simulation time	100 seconds
Nodes count	50
Data flow count	7 UDP (CBR)
Seeds used	10
Mobility	Waypoints in file

In these simulations, we introduce the OLSR (Optimized Link State Routing) protocol, it is a *proactive* routing protocole. As AODV is *reactive*, we find interesting to compare their respective performances with AAA under fading.

For these simulations, we study the following application layer statistics.

9.1 Bytes received

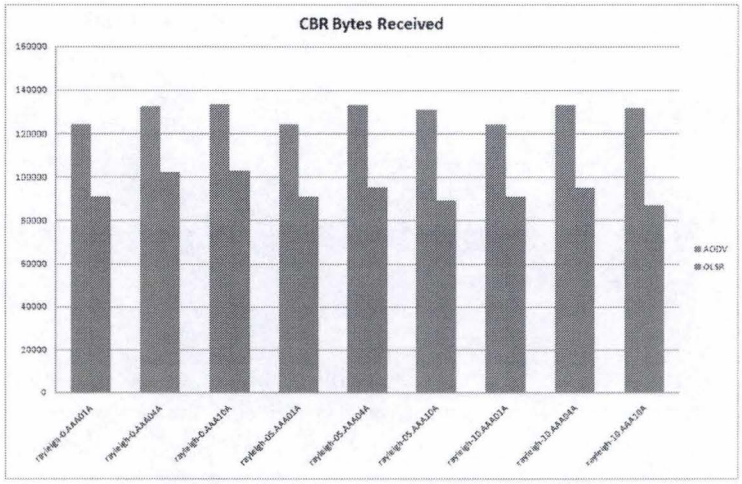

Figure 9.1: Application layer average bytes received

We can see that *AODV* protocol achieves larger data transfer than *OLSR*. AAA with 4 elements seems to be more efficient than antenna with 10 elements. This latter performs worse under fading with *OLSR* than an omnidirectional antenna. We have to notice that the omnidirectional antenna give always the same result even under fading as the gain is always $0dB$, we could use the QualNet fading implementation to compare result under fading.

9.2 Received throughput

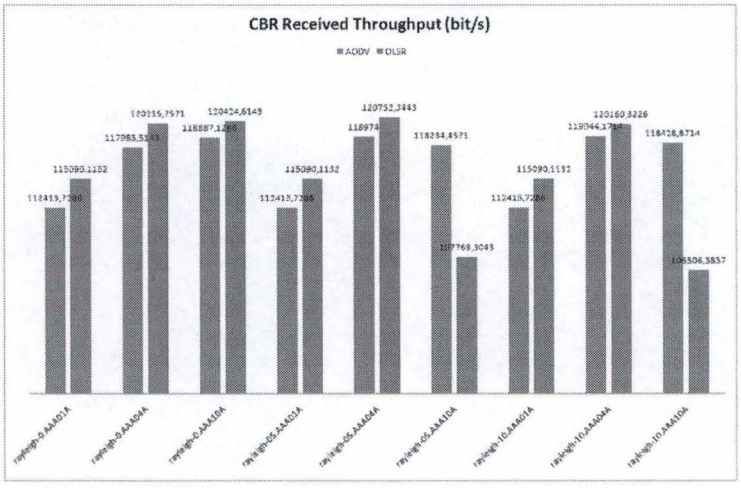

Figure 9.2: Application layer average received throughput

This chart shows the same behavior as in the previous one, except that *OLSR* is better than *AODV*. AAA with 10 elements are less efficient under fading with *OLSR* than a 4 elements antenna and even than an omnidirectional antenna.

9.3 End-to-end delay

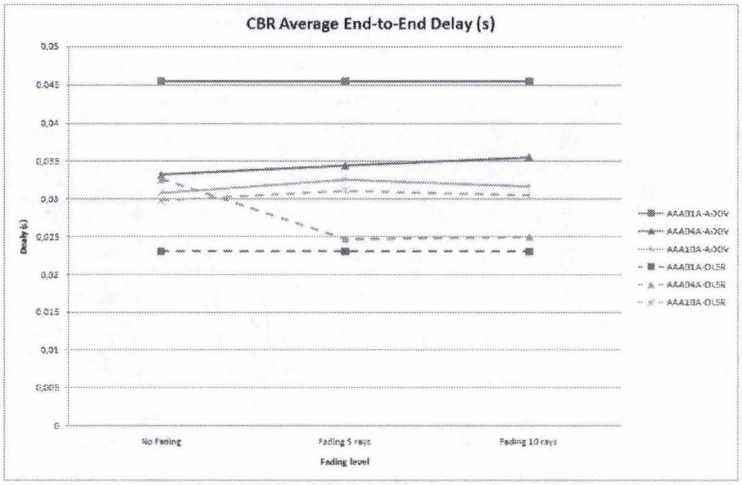

Figure 9.3: Application layer average end-to-end delay

OLSR, as a proactive routing protocol, causes lower delay in transmissions since the routes are already established. As in previous result charts, antennas with 10 elements are less efficient than other antennas. With *AODV*, delays are longer but the efficiency of the AAA according to the number of elements follows the expected behavior.

9.4 Conclusion

We only test one mobile network configuration (node count, node mobility and communications) and network layer configuration (default 802.11 CSMA/CA behavior). This is not enough representative to conclude on the performance of AODV vs OLSR. We still have some doubt on the integration of fading with omnidirectional antenna as the fading impact is null. This could be a part of a futur work with this AAA implementation.

Chapter 10

Conclusions

In this short chapter, we will remind the objectives of our project and review its current status. We also expose some perspectives for future work.

The first goal was to implement an AAA model in the QualNet simulator, to be able to simulate MANET behaviors where the devices use such antenna. This implementation was intended to be integrated in QualNet in a way that would allow other protocols to be tested or designed specifically for the AAA, with the possibility to measure the level of performances. This implementation can also be used for researches on protocol stack optimizations (cross-layer optimization).

The second goal was to measure the performances of routing protocols such as *AODV* or *OLSR*. This aspect was not thoroughly explored, and can be investigated in further research projects.

It is expected from the AAA to increase the performances through its ability to filter the incoming signals and to extract only the desired one from an environment with interference. Through this filtering ability, the AAA can counter physical phenomenons such as Rayleigh fading and Doppler effect.

It is important to mention that our implementation does not behave like a *real* AAA, since we considered certain simplifications :

- arrival time of different signals is not respected
- Wiener solution requires the reception to be over to compute the weight vector
- the problem of the preamble distribution is considered to be solved

To summarize the current status of our AAA model :

- Several parameters can be changed (see Section 4.1)
- Wiener solution implemented

- Rayleigh fading implemented
- Doppler effect implemented

A real AAA should use an iterative filtering, because the delay caused by the Wiener solution is not suited for real-time filtering. An implementation of the Ricean fading would increase the fading granularity to match a more realistic environment. These points can be a topic of future projects.

We also developped several tools to facilitate the preparation of batches, the execution of simulations and the gathering of results. Users guides will be written to facilitate the work of future researchers.

Bibliography

[1] Eric W. Weisstein. Argand diagram. From Mathworld, a Wolfram Web Resource.*http://mathworld.wolfram.com/ArgandDiagram.html.*

[2] Several unknown authors. Phase-shift keying, quadrature amplitude modulation. *Wikipedia, The Free Encyclopedia, http://en.wikipedia.org.* Date consulted : april 2007.

[3] N. Francois. A study of the fading effect on multi-hop wireless ad-hoc networks and its mitigation. Master's thesis, Facultés Universitaires Notre-Dame de la Paix, 2006.

[4] Y. Kamiya. Telecommunication engineering : Towards simulations on wireless ad hoc networks. Accelerated Cursus, 2006.

[5] Sang Bae Kaixin Xu, Mario Gerla. How effective is the ieee 802.11 rts/cts handshake in ad hoc networks? Technical report, University of California, Los Angeles, Computer Science Department, 2006.